Universidade do Minho
Escola de Engenharia

José Miguel Ribeiro Pinhão Pereira

Uma heurística de melhor encaixe para um problema de empacotamento a duas dimensões

outubro de 2013

José Miguel Ribeiro Pinhão Pereira **Uma heurística de melhor encaixe para um problema de empacotamento a duas dimensões**

UMinho | 2013

Universidade do Minho
Escola de Engenharia

José Miguel Ribeiro Pinhão Pereira

Uma heurística de melhor encaixe para um problema de empacotamento a duas dimensões

Dissertação de Mestrado
Mestrado em Engenharia de Sistemas

Trabalho realizado sob a orientação do
Professor Doutor Filipe Pereira Pinto da Cunha e Alvelos

outubro de 2013

Agradecimentos

Gostaria de expressar o meu agradecimento à Universidade do Minho, em particular à Escola de Engenharia, que me proporcionou os meios para realizar este trabalho.

De um modo especial ao Prof. Filipe Alvelos que me orientou neste projecto, de quem pude contar desde o primeiro momento com o seu apoio e solicitude constantes, mas também com um profundo sentido de profissionalismo que considero ter sido uma referência fundamental para o trabalho que realizei.

Ao Prof. Valério de Carvalho, Prof. Cláudio Alves e ao Prof. João Miguel Fernandes, pelo contributo e apoio que me deram no âmbito deste projecto.

Durante os últimos meses foram inúmeras as pessoas que, cientes do esforço necessário para a conciliação entre o início de uma carreira profissional em Lisboa e a finalização de uma etapa da minha formação académica em Braga, me animaram a levar a bom termo o objectivo que tinha entre mãos. O meu profundo agradecimento a todas elas, de um modo especial à minha família, sem a qual dificilmente teria perseverado.

Braga, Setembro de 2013.

José Miguel Ribeiro Pinhão Pereira

Este trabalho é dedicado à memória dos meus avós, Maria Cândida e José.

Esta dissertação foi desenvolvida no âmbito do projeto *"SearchCol - Metaheuristic search by column generation"* (PTDC/EIA-EIA/100645/2008), financiado por fundos nacionais através da Fundação para a Ciência e para a Tecnologia (PIDDAC) e co-financiado pelo Fundo Europeu de Desenvolvimento Regional (FEDER) através do COMPETE - Programa Operacional Factores de Competitividade (POFC).

Resumo

Nesta dissertação é apresentada uma heurística de melhor encaixe para um problema de empacotamento a duas dimensões. Este problema faz parte de um conjunto mais vasto de problemas de corte e empacotamento que são estudados em Investigação Operacional, e têm aplicação prática nas mais diversas áreas industriais.

São analisados diferentes modelos de empacotamento a duas dimensões. A heurística foi desenhada com base no modelo de melhor encaixe, e deu origem a uma ferramenta de *software* de optimização do planeamento de corte de matéria-prima. O desenvolvimento desta ferramenta foi inspirado por um caso real de uma empresa portuguesa de instalação de revestimentos, tendo como objectivo a optimização do processo de planeamento de corte de matéria-prima.

Foram realizados testes computacionais sobre a ferramenta desenvolvida tendo como referência um conjunto de instâncias reais, que permitiram aferir o nível qualitativo das soluções obtidas pela heurística face à qualidade das soluções reais existentes.

Palavras chave: Problemas de empacotamento a duas dimensões, Heurística de melhor encaixe.

Abstract

In this work, a best-fit heuristic for a two-dimensional bin packing problem is presented. This problem is part of a wider set of cutting and packing problems studied in the field of Operations Research, having multiple applications in various industrial environments.

A series of two-dimensional cutting and packing problems is analysed. Namely, level packing models, level and stack packing models and free cutting models. Furthermore, several resolution models are studied for the problems mentioned above.

The best-fit heuristic for two-dimensional bin packing problems was designed having the best-fit model as a starting point, and was integrated in the development of an industrial software tool for raw material cutting optimization. The development of the industrial tool was inspired by the real case of a portuguese surface industry.

The heuristic implemented was submitted to several software tests based on a set of real instances, that permitted to generate a benchmark capable of traducing the quality of the solutions obtained by the heuristic versus the initial existing results.

Keywords: Two-dimensional bin packing problems, Best-fit heuristic.

Índice

1 Introdução **19**

2 Revisão bibliográfica **23**

2.1 Problemas de corte e empacotamento 23

2.2 Problemas de empacotamento a duas dimensões 24

2.2.1 Empacotamento por níveis 26

2.2.2 Empacotamento por níveis e pilhas 27

2.2.3 Empacotamento livre 28

2.3 Modelos 29

2.3.1 Dois estágios 29

2.3.2 Dois estágios com rotação 35

2.3.3 Três estágios 39

2.4 Métodos heurísticos 44

2.4.1 Heurística *KP* - Itens orientados, corte guilhotinável 45

2.4.2 Heurística *KP* - Itens não orientados, corte guilhotinável 47

2.4.3 Heurística das Direcções Alternadas - Itens orientados, corte livre 48

2.4.4 Heurística do Perímetro em Contacto - Itens não orientados, corte livre 51

2.4.5 Heurística *HBP* - Itens orientados/não orientados, corte livre 55

2.4.6 Heurística Chão-tecto 60

3 Descrição do problema **63**

3.1 O caso de estudo *MPinto* 63

3.2 Problema de empacotamento a duas dimensões com minimização da altura da placa menos preenchida em altura 67

4 Método de solução **71**

4.1 Descrição 71

4.1.1 Heurística de melhor encaixe 71

4.1.2 Uma modificação da heurística de melhor encaixe 75

4.1.3 *Array Skyline* 76

4.2 Algoritmo 77

4.3 Exemplo 80

5 Implementação e resultados experimentais **85**

5.1 Implementação de modelos 2LBP e 2LBPr 85

5.2 Extensão de melhor encaixe 85

5.3 Resultados de testes a modelos 2LBP e 2LBPr 86

5.4 Heurística de melhor encaixe para um problema de empacotamento a duas dimensões 88

6 Conclusões e trabalho futuro **91**

7 Anexos **95**

Lista de Figuras

1 Esquema de empacotamento a duas dimensões em rolos versus empacotamento a duas dimensões em placas. 25
2 Empacotamento a duas dimensões por níveis. 27
3 Empacotamento a duas dimensões por níveis e pilhas. 28
4 Empacotamento livre. 29
5 Esquema do conjunto de itens da instância da Tabela 1. 32
6 Esquema do resultado de execução da instância da Tabela 1 pela heurística *TP*. 55
7 Estrutura do documento de preparação da *MPinto*. 64
8 Processo de alocação de itens da *MPinto*. 65
9 Função objectivo: Minimizar a altura da placa mais curta. Solução com pior qualidade. 69
10 Função objectivo: Minimizar a altura da placa mais curta. Solução com maior qualidade. 69
11 Exemplo de posicionamento junto ao vizinho mais alto. 73
12 Actualização do *array skyline*. 77
13 Políticas de posicionamento de itens. 78
14 Resultados possíveis da pesquisa de melhor encaixe. 79
15 Diagrama da heurística de melhor encaixe adaptada a problemas de empacotamento em placas. 79
16 Resultado da execução da instância para a extensão da heurística de melhor empacotamento com posicionamento à esquerda. 83
17 Resultado da execução da instância para a extensão da heurística de melhor encaixe com posicionamento ao lado do vizinho mais baixo, e posicionamento ao lado do vizinho mais alto. 84
18 Instância: "Quantidades" - *MPinto*. 97
19 Instância: "Quantidades" - *2LBP*. 99
20 Instância: "Quantidades" - *best-fit 2BP*. 101

Lista de Tabelas

1 Instância de um problema 2BP 32

2 Instância 2 36

3 Instância 2 - Criação de itens duplicados com rotação de 90^{o} 37

4 Instância 2 - Itens ordenados por valores de altura não crescentes 37

5 Instância 2 - Correspondência entre índices dos itens originais e itens duplicados. 38

6 Cálculo dos *scores* para a instância da Tabela 1 na heurística *TP*, com placas de tamanho $H = 8$ e $W = 10$. 55

7 Instância para a extensão à heurística de melhor empacotamento 80

8 Orientação horizontal e ordenação de itens. 81

9 Resultados de execução do modelo 2LBP 88

10 Resultados de execução do modelo 2LBPr 88

11 Resultados da heurística de melhor encaixe para um problema de empacotamento a duas dimensões 89

Lista de Abreviaturas

1BP - Empacotamento a uma dimensão em objectos de dimensões finitas.

2BP - Empacotamento a duas dimensões em objectos de dimensões finitas.

2LBP - Modelo de empacotamento a duas dimensões, guilhotinável a duas fases, com itens orientados.

2LBPr - Modelo de empacotamento a duas dimensões, guilhotinável a duas fases, com itens não orientados.

AD - Heurística de empacotamento das direcções alternadas.

bins - Sinónimo de placas. Objectos com dimensões finitas.

C&P - Corte e empacotamento (*Cutting & Packing*).

FC - Heurística de empacotamento *Floor-Ceiling* (Chão-tecto).

FO - Função objectivo.

H - Altura da placa.

h - Altura do item.

KP - Problema de empacotamento de *Knapsack*.

SBSBPP - *Single Bin Size Bin Packing Problem*.

SP - Empacotamento em um objecto com largura finita e altura infinita, ou vice-versa.

strip - Objecto com largura finita e altura infinita, ou vice-versa.

TP - Problema de empacotamento do perímetro em contacto.

TI - Tecnologias de informação.

W - Largura da placa.

w - Largura do item.

1 Introdução

Nesta dissertação estuda-se um problema de empacotamento a duas dimensões. Este problema faz parte de um conjunto mais vasto de problemas de corte e empacotamento que são estudados em Investigação Operacional, e têm aplicação prática nas mais diversas áreas industriais.

Um problema de corte e empacotamento consiste em alocar um conjunto de itens a um conjunto de objectos. Apesar de partilharem uma estrutura lógica comum, os problemas de corte diferem dos problemas de empacotamento nas suas áreas de aplicação.

O início da investigação sobre os problemas de corte e empacotamento remonta a meados do século XX. O estudo nesta área vai-se desenvolvendo ao longo dos anos sem que surjam normalizações capazes de definir as fronteiras das diferentes vertentes de investigação existentes. No início dos anos 90, surge através de Dyckoff [3] a primeira tipologia de problemas de corte e empacotamento, criada para uniformizar a classificação de problemas e organizar o trabalho realizado pela comunidade científica nesta área. Anos mais tarde são propostas tipologias alternativas, com o propósito de adaptar a classificação de problemas à evolução do estudo na área, e de eliminar ambiguidades identificadas na tipologia existente, como é o caso do trabalho realizado por Wäscher et al. [11] .

Enquanto as tipologias de empacotamento fornecem um conhecimento geral sobre as diferentes classes de problemas existentes, os estudos realizados por Lodi et al. [8], [7], [9] fornecem um estudo sobre um vasto conjunto de modelos e métodos de solução para problemas de corte e empacotamento, quer heurísticos, quer determinísticos. Nesta dissertação são analisados alguns destes modelos, em particular, modelos para problemas de corte e empacotamento a duas dimensões.

A heurística de melhor encaixe para um problema de corte e empacotamento a duas dimensões foi construída com base na adaptação de um dos modelos analisados: a heurística de melhor encaixe. O modelo desenhado foi implementado numa solução de *software* com aplicação prática em diferentes áreas industriais, embora tenha sido inspirada num caso prático real de uma empresa de instalação de revestimentos portuguesa, a *MPinto*.

O objectivo fundamental do presente trabalho, é reduzir os custos associados ao processo produtivo da empresa *MPinto*. A optimização que se pretende desenvolver, visa uma melhoria dos níveis de produtividade da empresa, através de um compromisso entre a redução do número de horas-homem envolvido no processo de preparação de obra versus o nível de consumo de matéria-prima em cada projecto.

Outro objectivo deste projecto é assegurar que as soluções obtidas para o processo de corte são geradas de forma sistemática e objectiva, reduzindo o máximo de factores entregues à arbitrariedade do colaborador, garantindo a constância e coerência dos resultados. Através da construção de um modelo de corte e empacotamento para o caso *MPinto*, não se pretende modelar e replicar o método de alocação utilizado, mas sim construir um modelo capaz de fornecer soluções com qualidade satisfatória e reduzir drasticamente o seu tempo de execução.
O processo de planeamento de corte a duas dimensões no âmbito da indústria de revestimentos de superfícies envolve um elevado número de horas-homem quando é levado a cabo de forma manual, ainda que auxiliado por ferramentas TI. A sustentabilidade de empresas deste ramo num mercado altamente competitivo à escala global, vê-se ameaçada pela diferença do custos associados a mão-de-obra em outros países e, por outro lado, pelo investimento em tecnologia capaz de optimizar a produção realizada por empresas concorrentes.
Revela-se vital a necessidade de investimento na optimização do processo produtivo, capaz de se traduzir em:

- Redução do tempo gasto em todas as fases do projecto, desde a fase de planeamento de obra até à fase de implementação;
- Redução da quantidade de matéria-prima gasta na aplicação de revestimentos no local de obra;
- Aumento da produtividade dos colaboradores afectos a todas as fases da produção, com especial incidência sobre o planeamento e na instalação do revestimento no local de obra;
- Minimização dos custos associados ao processo produtivo;
- Maximizar o reaproveitamento de matéria-prima desperdiçada em projectos realizados;
- Assegurar a coerência das metodologias aplicadas em projectos e maximizar níveis de qualidade do produto final entregue ao cliente.

A utilização de uma ferramenta de planeamento de empacotamento a duas dimensões automático, como é o caso da ferramenta desenvolvida no âmbito do presente projecto, potencia uma melhoria da produtividade e da competitividade da empresa, aproximando-a dos objectivos acima delineados.
Esta dissertação está organizada em seis capítulos. No capítulo 2 é feita uma revisão bibliográfica sobre problemas de corte e empacotamento. No capítulo 3 é introduzido o problema

abordado nesta dissertação. No capítulo 4 é descrito o método de solução desenvolvido. No capítulo 5 descreve-se a implementação efectuada e os resultados obtidos. No último capítulo, apresentam-se as conclusões do trabalho realizado e indicam-se possíveis caminhos para a sua continuação.

2 Revisão bibliográfica

2.1 Problemas de corte e empacotamento

Os problemas de corte e empacotamento são uma área de estudo da Investigação Operacional, que se encontram presentes em diversos tipos de processos industriais. Apesar de se distinguirem entre si nas suas aplicações práticas, problemas de corte e problemas de empacotamento são estudados em conjunto por partilharem uma estrutura lógica comum. Por razões práticas, de ora em diante problemas de corte e empacotamento serão referidos como problemas de empacotamento.

É possível identificar inúmeros casos de problemas de empacotamento no ambiente armazenista, de um modo concreto no processo de alocação de um conjunto de componentes numa superfície ou no processo de carregamento de paletes de produtos para um veículo de transporte de mercadorias. Como exemplo de problemas de corte podemos considerar o processo de corte de figuras geométricas a partir de placas de madeira com iguais dimensões. Este tipo de problemas é encontrado nas indústrias de plásticos, metalúrgica, vidreira, madeireira, têxtil, papeleira, entre outras.

Um problema de empacotamento é definido como um processo de alocação de um conjunto de itens a um conjunto de objectos. Os dois conjuntos de elementos, itens e objectos, são caracterizados funcionalmente de forma distinta. Os itens consistem no resultado esperado da resolução do problema de empacotamento a que o conjunto de objectos é submetido, ou seja, os itens são considerados o produto final enquanto os objectos são os recursos.

Os elementos do conjunto de objectos e de itens podem ser definidos em uma, duas, três ou mais dimensões geométricas. O número de dimensões geométricas é determinante na classificação do problema de empacotamento considerado. O processo de organização de livros numa estante é um exemplo de um problema de empacotamento a três dimensões, e a organização da capa de um jornal poderá ser considerado um problema de empacotamento a duas dimensões. Na presente dissertação são analisados problemas em que os itens e objectos são caracterizados por duas dimensões.

O processo de alocação consiste em seleccionar uma parte ou a totalidade dos itens, agrupando-os em um ou mais subconjuntos (através da identificação de padrões por combinações geométricas) e alocando cada um dos subconjuntos gerados a um objecto, garantindo a observação das seguintes propriedades:

- A totalidade dos elementos que integram os subconjuntos de itens seleccionados tem dimensões compatíveis com as dimensões do objecto onde são alocados. Assim sendo, as dimensões de cada item não ultrapassam as dimensões do respectivo objecto.

- Todos os itens alocados num determinado objecto não se sobrepõem entre si. Uma solução válida para um problema de empacotamento poderá resultar na utilização de uma parte ou da totalidade dos objectos disponíveis, e de uma parte ou da totalidade do conjunto de itens.

2.2 Problemas de empacotamento a duas dimensões

Os problemas de empacotamento a duas dimensões são uma classe de problemas de empacotamento onde objectos e itens são caracterizados por duas dimensões geométricas. Podemos definir um problema de empacotamento a duas dimensões da seguinte forma:
Dado um conjunto finito não nulo J com n itens rectangulares $j \in J = \{1, \ldots, n\}$ em que cada item é caracterizado pelas dimensões (w_j, h_j), respectivamente largura e altura, e considerando um conjunto finito não nulo I com m objectos rectangulares $i \in I = \{1, \ldots, m\}$ de dimensões (W_i, H_i), o objectivo do problema é empacotar o máximo de itens na menor quantidade de objectos possível.
Nos casos em que os objectos têm dimensões finitas, estamos perante problemas de empacotamento a duas dimensões em placas. No caso particular em que o conjunto de objectos é constituído por apenas um elemento com largura W e altura infinita ($H = \infty$), trata-se de um problema de empacotamento a duas dimensões em rolos (*Two dimensional strip packing problem*). Neste caso, o objectivo do problema é empacotar todos os itens utilizando a menor altura do rolo possível, ou seja, minimizando o valor de H. Na Figura 1 é possível observar um esquema que representa as diferenças entre problemas de empacotamento a duas dimensões em placas e em rolos.

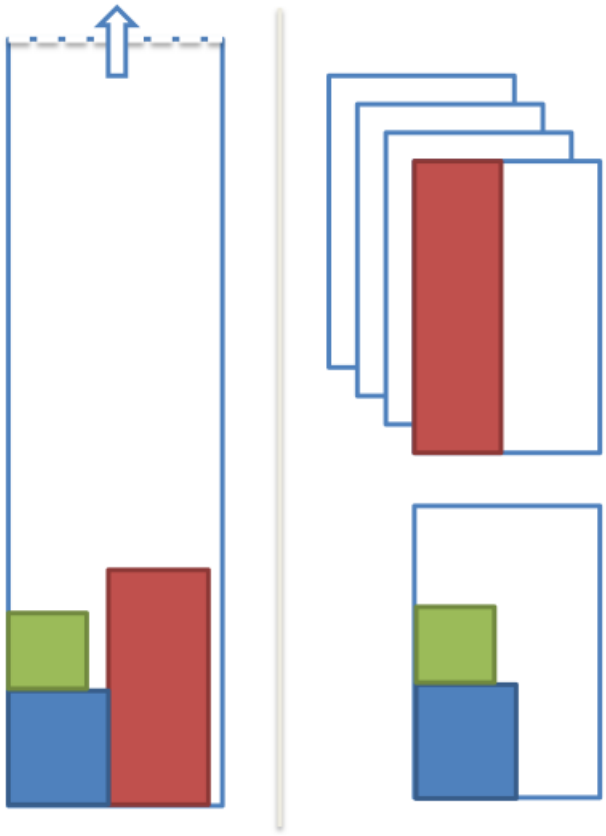

Figura 1: Esquema de empacotamento a duas dimensões em rolos versus empacotamento a duas dimensões em placas.

Na versão clássica de problemas de empacotamento em placas de igual tamanho, o objectivo é empacotar a totalidade dos itens no mínimo de placas possíveis. Este problema é classificado como 2/V/I/M, segundo a tipologia proposta por Dyckoff [3]. O tipo de problemas 2/V/I/M é definido pela seguintes propriedades:

- Dimensionalidade: 2 – O problema é bi-dimensional, o que significa que os itens têm duas dimensões.
- Tipo de tarefa: V – Todos os itens têm que ser combinados em padrões que serão posteriormente alocados a uma selecção de objectos. (O símbolo V é uma referência ao termo alemão *Verladeproblem*, usado para problemas em que todos os itens são alocados num subconjunto de objectos).
- Classificação de objectos: I – Objectos idênticos, isto é, todos os objectos têm as mesmas dimensões.

- Classificação de itens: M – Observa-se uma grande heterogeneidade de dimensões no conjunto de itens.

Dyckoff [3] refere-se a esta classificação como o problema clássico de empacotamento a duas dimensões em placas (*Classical Bin Packing Problem*). Wäscher et al. [11] considerando que esta tipologia era demasiado ambígua, propuseram uma classificação alternativa para a mesma classe na sua tipologia associando-a a um tipo de problemas intermédio denominado SBSBPP (*Single Bin Size Bin Packing Problem*), que é traduzido por problema de empacotamento em placas de dimensões iguais. A classificação alternativa é baseada em propriedades ligeiramente diferentes da classificação anterior:

- Dimensionaliade: Bi-dimensional.
- Tipo de tarefa: Minimização de input.
- Classificação de objectos: Objectos idênticos.
- Classificação de itens: Itens com dimensões muito diferentes.

Tanto os itens como os objectos podem adquirir dimensões diferentes. A diversidade de dimensões existente no conjunto de itens e objectos, assim como a sua cardinalidade, são factores que determinam a classificação do tipo de problema de empacotamento analisado. Para além destes factores, também as restrições sobre a orientação dos itens no processo de alocação (capacidade de rotação de itens na alocação), restrições sobre posicionamento ortogonal e de corte transversal, entre outras, determinam a classificação do problema, a selecção do modelo a ser utilizado na sua resolução e a qualidade da mesma.
Em seguida, analisamos três diferentes modos de empacotamento, que estão na base dos modelos estudados na presente dissertação.

2.2.1 Empacotamento por níveis

O empacotamento por níveis é um método de alocação ortogonal de itens, que garante a possibilidade de corte transversal. Ou seja, o plano de empacotamento do objecto é divisível de ponta a ponta sem causar colisões com qualquer item. Um exemplo real de aplicação do método de empacotamento por níveis é o caso do processo de corte de rolo de papel através de uma guilhotina automática, em que a lâmina tem necessariamente que cobrir transversalmente o objecto em cada corte. A divisão das áreas ocupadas pelos itens no objecto é realizado em

duas fases. Na primeira fase são realizadas as divisões horizontais que produzem os níveis, e na segunda fase as divisões verticais separando os itens empacotados em cada nível. Na Figura 2 é possível observar um esquema de um problema de empacotamento por níveis.

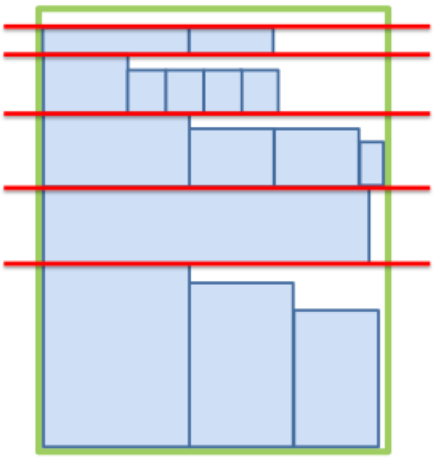

Figura 2: Empacotamento a duas dimensões por níveis.

2.2.2 Empacotamento por níveis e pilhas

O empacotamento por níveis e pilhas é semelhante ao empacotamento por níveis, com a particularidade de adicionar uma fase adicional de empacotamento, possibilitando no mesmo nível ter pilhas verticais de itens. Apresenta a vantagem de possibilitar um aproveitamento do espaço vertical em cada nível de empacotamento, diminuindo o desperdício gerado no empacotamento por níveis. A melhoria da qualidade das soluções geradas tem como revés a maior complexidade associada a este tipo de problemas, estando dependentes da cardinalidade e da diversidade de dimensões no conjunto de itens e objectos. O esquema da Figura 3 representa um problema de empacotamento por níveis e pilhas.

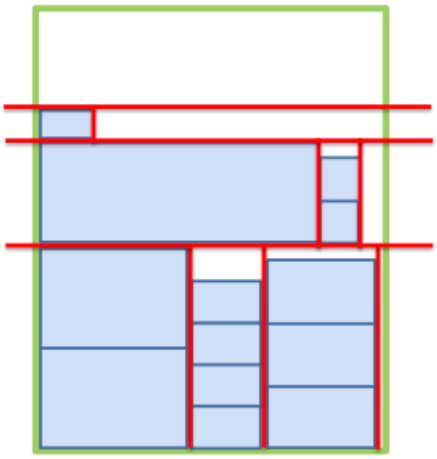

Figura 3: Empacotamento a duas dimensões por níveis e pilhas.

Os critérios considerados para a criação de pilhas de itens varia de modelo para modelo. Não sendo invariavelmente necessário que os itens de cada pilha tenham igual largura, é conveniente que tenham largura aproximada para o aproveitamento do espaço vertical dentro do nível de empacotamento.

2.2.3 Empacotamento livre

Ao contrário dos tipos de empacotamento anteriores, no empacotamento livre não existem quaisquer restrições de alocação, à excepção das propriedades básicas de empacotamento já enunciadas: a não sobreposição de itens, e a compatibilidade de dimensões de itens e de objectos.

O empacotamento livre é frequentemente resolvido por métodos heurísticos, onde o objectivo é encontrar uma solução em tempo útil, o mais aproximadamente possível do valor óptimo. O esquema da Figura 4 representa um problema de empacotamento livre.

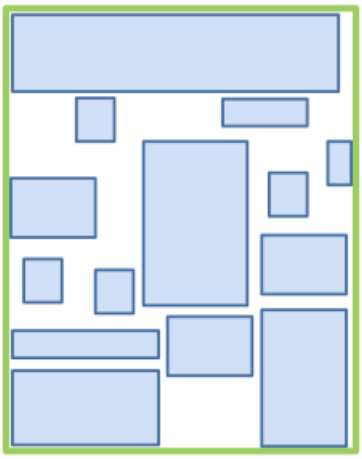

Figura 4: Empacotamento livre.

2.3 Modelos

2.3.1 Dois estágios

Lodi et al. [7] introduziram um modelo de programação inteira para empacotamento por níveis: o modelo 2LBP. O modelo 2LBP é considerado um modelo de dois estágios, sendo que o corte dos objectos é processado em duas fases distintas: na primeira fase são realizados cortes horizontais de ponta a ponta definindo tiras na placa, e na segunda fase são realizados cortes verticais entre os itens alocados em cada tira gerada na primeira fase.
O primeiro nível de cada placa localiza-se sempre na sua base inferior. Os itens são empacotados alinhados à esquerda, com a base encostada à base do respectivo nível. A posição de um nível seguinte é determinada pela linha horizontal traçada pelo item mais alto empacotado no nível inferior.
Uma solução para o modelo 2LBP obedece às seguintes propriedades:

1. Para cada nível de corte, o item que se encontra mais à esquerda é o mais alto.
2. Para cada placa, o nível que se encontra na posição mais abaixo é o mais alto.
3. Os itens são ordenados e numerados por valores de altura h_j não crescentes.

Cada nível de corte é inicializado e identificado pelo item que se encontra na posição extrema do lado esquerdo, e cada placa é inicializada e identificada pelo nível que se encontra na alocado na sua base.

O modelo de Lodi et al. [7] baseia-se na definição de n níveis possíveis, sendo cada nível i inicializado pelo item i, e na definição de n placas possíveis, sendo cada placa inicializada por um possível nível.

Assumindo que

- J corresponde ao conjunto de itens do problema de empacotamento;
- n corresponde ao número de elementos do conjunto J;
- w_i, h_i são respectivamente as medidas de largura e altura do item i, $i \in J$;
- W e H são respectivemente as medidas de largura e altura das placas;

as variáveis de decisão do modelo 2LBP são:

$$y_i = \begin{cases} 1, & \text{se e só se o item } i \text{ inicializa o nível } i, \text{ onde } i \in J. \\ 0, & \text{caso contrário.} \end{cases}$$

$$q_k = \begin{cases} 1, & \text{se e só se o nível } k \text{ (inicializado pelo item } k) \text{ inicializa a placa } k, \text{ onde } k \in J. \\ 0, & \text{caso contrário.} \end{cases}$$

$$x_{ij} = \begin{cases} 1, & \text{se e só se o item } j \text{ for empacotado no nível } i \text{ (inicializado pelo item } i), \\ & \text{onde } i, j \in J. \\ 0, & \text{caso contrário.} \end{cases}$$

$$z_{ki} = \begin{cases} 1, & \text{se e só se o nível } i \text{ (inicializado pelo item } i) \text{ for empacotado na placa } k \\ & \text{(inicializada pelo nível } k \text{ com o item } k), \text{ onde } k, i \in J. \\ 0, & \text{caso contrário.} \end{cases}$$

A partir das variáveis de decisão descritas acima, é possível construir o seguinte modelo de programação inteira para o problema 2LBP:

$$Min \sum_{k=1}^{n} q_k \tag{1}$$

sujeito a:

$$\sum_{i=1}^{j-1} x_{ij} + y_j = 1 \qquad (j = 1...n) \tag{2}$$

$$\sum_{j=i+1}^{n} w_j x_{ij} \leq (W - w_i) y_i \qquad (i = 1...n-1) \tag{3}$$

$$\sum_{k=1}^{i-1} z_{ki} + q_i = y_i \qquad (i = 1...n) \tag{4}$$

$$\sum_{i=k+1}^{n} h_i z_{ki} \leq (H - h_k) q_k \qquad (k = 1...n-1) \tag{5}$$

$$y_i, x_{ij}, q_k, z_{ki} \in \{0, 1\} \; \forall i, j, k \tag{6}$$

Função objectivo (1): A expressão (1) representa a função objectivo do modelo. Esta função minimiza o número de placas usadas numa solução do modelo.

Restrições (2): Garantem que todos os itens são empacotados (inicializando um nível ou sendo empacotados num nível inicializado por outro item).

Restrições (3): As restrições (3) garantem que um item só pode ser empacotado num nível inicializado e que a largura total dos itens empacotados num nível não excede a largura da placa.

Restrições (4): As restrições (4) garantem que todos os níveis são empacotados (inicializando uma placa ou sendo empacotados numa placa inicalizada por outro nível).

Restrições (5): Estas restrições garantem que um nível só pode ser empacotado numa placa inicializada, e que a altura somada dos itens que inicializam os níveis empacotados numa placa

não excede a altura da mesma.

Restrições (6): Define o domínio das variáveis de decisão.

Para possibilitar uma melhor compreensão do modelo de programação inteira apresentado acima, apresentamos um exemplo do modelo para a instância da Tabela 1, ilustrada na Figura 5.

Tabela 1: Instância de um problema 2BP

Indíce i	Código	Largura w_i	Altura h_i
1	A	4	6
2	B	4	4
3	C	3	4
4	D	8	3
5	E	1	3
6	F	6	2
7	G	2	2
8	H	4	1

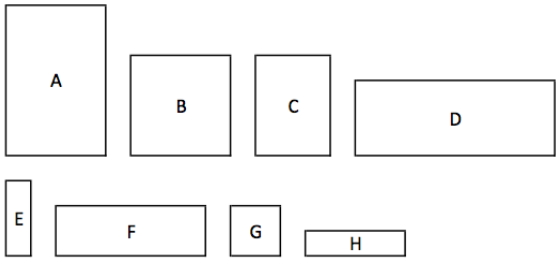

Figura 5: Esquema do conjunto de itens da instância da Tabela 1.

Assumindo que as dimensões das placas são $H = 20$ e $W = 10$, o modelo correspondente à

instância da Tabela 1 pode ser definido da seguinte forma:

$$Min\ q_1 + q_2 + q_3 + q_4 + q_5 + q_6 + q_7 + q_8 \quad (FO)$$

$sujeito\ a:$

$$y_1 = 1 \quad (R1)$$

$$x_{12} + y_2 = 1 \quad (R2)$$

$$x_{13} + x_{23} + y_3 = 1 \quad (R3)$$

$$x_{14} + x_{24} + x_{34} + y_4 = 1 \quad (R4)$$

$$x_{15} + x_{25} + x_{35} + x_{45} + y_5 = 1 \quad (R5)$$

$$x_{16} + x_{26} + x_{36} + x_{46} + x_{56} + y_6 = 1 \quad (R6)$$

$$x_{17} + x_{27} + x_{37} + x_{47} + x_{57} + x_{67} + y_7 = 1 \quad (R7)$$

$$x_{18} + x_{28} + x_{38} + x_{48} + x_{58} + x_{68} + x_{78} + y_8 = 1 \quad (R8)$$

$$4x_{12} + 3x_{13} + 8x_{14} + 1x_{15} + 6x_{16} + 2x_{17} + 4x_{18} \leq (10 - 4)y_1 \quad (R9)$$

$$3x_{23} + 8x_{24} + 1x_{25} + 6x_{26} + 2x_{27} + 4x_{28} \leq (10 - 4)y_2 \quad (R10)$$

$$8x_{34} + 1x_{35} + 6x_{36} + 2x_{37} + 4x_{38} \leq (10 - 3)y_3 \quad (R11)$$

$$1x_{45} + 6x_{46} + 2x_{47} + 4x_{48} \leq (10 - 8)y_4 \quad (R12)$$

$$6x_{56} + 2x_{57} + 4x_{58} \leq (10 - 1)y_5 \quad (R13)$$

$$2x_{67} + 4x_{68} \leq (10 - 6)y_6 \quad (R14)$$

$$4x_{78} \leq (10 - 2)y_7 \quad (R15)$$

$$q_1 = y_1 \tag{R16}$$

$$z_{12} + q_2 = y_2 \tag{R17}$$

$$z_{13} + z_{23} + q_3 = y_3 \tag{R18}$$

$$z_{14} + z_{24} + z_{34} + q_4 = y_4 \tag{R19}$$

$$z_{15} + z_{25} + z_{35} + z_{45} + q_5 = y_5 \tag{R20}$$

$$z_{16} + z_{26} + z_{36} + z_{46} + z_{56} + q_6 = y_6 \tag{R21}$$

$$z_{17} + z_{27} + z_{37} + z_{47} + z_{57} + z_{67} + q_7 = y_7 \tag{R22}$$

$$z_{18} + z_{28} + z_{38} + z_{48} + z_{58} + z_{68} + z_{78} + q_8 = y_8 \tag{R23}$$

$$4z_{12} + 4z_{13} + 3z_{14} + 3z_{15} + 2z_{16} + 2z_{17} + 1z_{18} \leq (20-6)q_1 \tag{R24}$$

$$4z_{23} + 3z_{24} + 3z_{25} + 2z_{26} + 2z_{27} + 1z_{28} \leq (20-4)q_2 \tag{R25}$$

$$3z_{34} + 3z_{35} + 2z_{26} + 2z_{37} + 1z_{38} \leq (20-4)q_3 \tag{R26}$$

$$3z_{45} + 2z_{46} + 2z_{47} + 1z_{48} \leq (20-3)q_4 \tag{R27}$$

$$2z_{56} + 2z_{57} + 1z_{58} \leq (20-3)q_5 \tag{R28}$$

$$2z_{67} + 1z_{68} \leq (20-2)q_6 \tag{R29}$$

$$1z_{78} \leq (20-2)q_7 \tag{R30}$$

$$q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8 \in \{0,1\} \tag{R31}$$

$$z_{12}, z_{13}, z_{14}, z_{15}, z_{16}, z_{17}, z_{18}, z_{23}, z_{24}, z_{25}, z_{26}, z_{27}, z_{28}, z_{34}, z_{35}, z_{36}, z_{37}, z_{38},$$
$$z_{45}, z_{46}, z_{47}, z_{48}, z_{56}, z_{57}, z_{58}, z_{67}, z_{68}, z_{78} \in \{0,1\} \tag{R32}$$

$$y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8 \in \{0,1\} \tag{R33}$$

$$x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{23}, x_{24}, x_{25}, x_{26}, x_{27}, x_{28}, x_{34}, x_{35}, x_{36}, x_{37}, x_{38},$$
$$x_{45}, x_{46}, x_{47}, x_{48}, x_{56}, x_{57}, x_{58}, x_{67}, x_{68}, x_{78} \in \{0,1\} \tag{R34}$$

2.3.2 Dois estágios com rotação

No modelo 2LBP, os itens são alocados com orientação fixa. Lodi et al. [9] apresentaram uma variante do modelo 2LBP em que os itens podem alterar a sua orientação. Nesta variante, para cada item a empacotar, é criado um item duplicado com uma rotação de $90^{\underline{o}}$. Deste modo, o item duplicado δ_j corresponde à rotação do item j (onde $j \in \{1...n\}$) tem dimensões $w_{\delta_j} = h_j$ e $h_{\delta_j} = w_j$.

O modelo de programação inteira para a extensão 2LBPr é definido da seguinte forma:

$$Min \sum_{k=1}^{2n} q_k \tag{7}$$

sujeito a:

$$\sum_{i=1}^{j-1} x_{ij} + y_j \leq 1 \qquad (j = 1...2n) \tag{8}$$

$$\sum_{j=i+1}^{n} w_j x_{ij} \leq (W - w_i) y_i \qquad (i = 1...2n-1) \tag{3}$$

$$\sum_{k=1}^{i-1} z_{ki} + q_i = y_i \qquad (i = 1...2n) \tag{4}$$

$$\sum_{i=k+1}^{n} h_i z_{ki} \leq (H - h_k) q_k \qquad (k = 1...2n-1) \tag{5}$$

$$y_j + \sum_{i=1}^{j-1} x_{ij} + y_{\delta_j} + \sum_{i=1}^{\delta_j - 1} x_{i\delta_j} = 1 \qquad \forall j \in I, j < \delta_j \tag{9}$$

$$y_i, x_{ij}, q_k, z_{ki} \in \{0, 1\} \; \forall i, j, k \tag{6}$$

O número de itens que o modelo 2LBPr recebe corresponde ao dobro do número de itens recebidos pelo modelo 2LBP, uma vez que o conjunto de itens para o modelo 2LBPr contêm também os itens duplicados. Desta forma, não havendo alterações à indexação dos itens atribuída, é possível assumir que o item duplicado $\delta_i = i + n$, para qualquer item i.
O modelo 2LBPr difere do modelo 2LBP nas expressões (7), (8) e (9):

Função Objectivo (7): A função objectivo para o modelo 2LBPr difere da expressão (1)

apenas no número de itens recebidos ($2n$), uma vez que para cada item $i \in n$ é associado um item i_r correspondente à sua rotação.

Restrição (8): Para cada par de itens i e j (onde $i < j$) pertencentes ao conjunto de itens J, ou o item j é empacotado no nível i (inicializado pelo item i), ou o item j inicializa o nível j, ou então não é empacotado.

Restrição (9): Para cada item $j \in J$, ou j inicializa o nível y_j, ou j é empacotado num nível já existente, ou o item δ_j (duplicado de j com rotação de 90^{o}) inicializa o nível y_{δ_j} ou então δ_j é empacotado num nível já existente. Esta restrição garante que os itens j e δ_j não são empacotados simultaneamente.

As restrições (3,4,5,6) são as mesmas que no modelo 2LBP.

Consideremos agora uma nova instância de problemas 2BP, com os três primeiros itens da instância anterior apresentada na Tabela 1:

Tabela 2: Instância 2

Índice i	Código c_i	Largura w_i	Altura h_i
1	A	4	6
2	B	4	4
3	C	3	4

Seja J o conjunto de itens apresentado na Tabela 2, onde $i \in \{1, 2, 3\}$, item $i \in J$. Para cada item $i \in J$, é criado um item duplicado δ_i com uma rotação de 90^{o}. Consequentemente, $w_{\delta_i} = h_i,\ h_{\delta_{i=w_i}}$.

Antes do processamento do modelo 2LBPr, todos os itens são ordenados por valores não crescentes de altura. Após a criação dos itens duplicados é mantido um registo de correspondência entre os índices dos itens originais e os índices dos itens duplicados, antes do processo de ordenação por alturas:

Na Tabela 3 são replicados os itens da Tabela 2, com uma rotação de 90^{o}, e na Tabela 4 os itens são ordenados por valores não crescentes de altura. Na Tabela 5 é registada a correspondência dos índices entre itens originais e itens duplicados com rotação de 90^{o}.

Tabela 3: Instância 2 - Criação de itens duplicados com rotação de 90º

Índice i	Código c_i	Largura w_i	Altura h_i
1	A	4	6
2	B	4	4
3	C	3	4
4	Ar	6	4
5	Br	4	4
6	Cr	4	3

Tabela 4: Instância 2 - Itens ordenados por valores de altura não crescentes

Índice i	Índice antigo	Código c_i	Largura w_i	Altura h_i
1	6	Ar	6	4
2	1	A	4	6
3	2	B	4	4
4	7	Br	4	4
5	8	Cr	4	3
6	3	C	3	4

Após a ordenação dos itens, é possível construir o modelo 2LBPr para a instância 2, segundo a informação da Tabela 4 e assumindo que objectos têm dimensões $H = 20$ e $W = 10$:

$$Min\ q_1 + q_2 + q_3 + q_4 + q_5 + q_6 \qquad (FO)$$

$sujeito\ a$:

$$y_1 \leq 1 \qquad (R1)$$

$$x_{12} + y_2 \leq 1 \qquad (R2)$$

$$x_{13} + x_{23} + y_3 \leq 1 \qquad (R3)$$

$$x_{14} + x_{24} + x_{34} + y_4 \leq 1 \qquad (R4)$$

$$x_{15} + x_{25} + x_{35} + x_{45} + y_5 \leq 1 \qquad (R5)$$

$$x_{16} + x_{26} + x_{36} + x_{46} + x_{56} + y_6 \leq 1 \qquad (R6)$$

Tabela 5: Instância 2 - Correspondência entre índices dos itens originais e itens duplicados.

Índice	Índice original	Índice duplicado
A	2	1
B	3	4
C	6	5

$$4x_{12} + 4x_{13} + 4x_{14} + 4x_{15} + 3x_{16} \leq (10 - 6)y_1 \quad (R7)$$

$$4x_{23} + 4x_{24} + 4x_{25} + 3x_{26} \leq (10 - 4)y_2 \quad (R8)$$

$$4x_{34} + 4x_{35} + 3x_{36} \leq (10 - 4)y_3 \quad (R9)$$

$$4x_{45} + 3x_{46} \leq (10 - 4)y_4 \quad (R10)$$

$$3x_{56} \leq (10 - 4)y_5 \quad (R11)$$

$$q_1 = y_1 \quad (R12)$$

$$z_{12} + q_2 = y_2 \quad (R13)$$

$$z_{13} + z_{23} + q_3 = y_3 \quad (R14)$$

$$z_{14} + z_{24} + z_{34} + q_4 = y_4 \quad (R15)$$

$$z_{15} + z_{25} + z_{35} + z_{45} + q_5 = y_5 \quad (R16)$$

$$z_{16} + z_{26} + z_{36} + z_{46} + z_{56} + q_6 = y_6 \quad (R17)$$

$$6z_{12} + 4z_{13} + 4z_{14} + 3z_{15} + 4z_{16} \leq (20 - 4)q_1 \quad (R18)$$

$$4z_{23} + 4z_{24} + 3z_{25} + 4z_{26} \leq (20 - 6)q_2 \quad (R19)$$

$$4z_{34} + 3z_{35} + 4z_{26} \leq (20 - 4)q_3 \quad (R20)$$

$$3z_{45} + 4z_{46} \leq (20 - 4)q_4 \quad (R21)$$

$$4z_{56} \leq (20 - 3)q_5 \quad (R22)$$

$$y_1 + y_2 + x_{12} = 1 \qquad (R23)$$

$$y_3 + x_{13} + x_{23} + y_4 + x_{14} + x_{24} + x_{34} = 1 \qquad (R24)$$

$$y_5 + x_{15} + x_{25} + x_{35} + x_{45} + y_6 + x_{16} + x_{26} + x_{36} + x_{46} + x_{56} = 1 \qquad (R25)$$

$$q_1, q_2, q_3, q_4, q_5, q_6 \in \{0, 1\} \qquad (R26)$$

$$z_{12}, z_{13}, z_{14}, z_{15}, z_{16}, z_{23}, z_{24}, z_{25}, z_{26}, z_{34}, z_{35}, z_{36},$$
$$z_{45}, z_{46}, z_{56} \in \{0, 1\} \qquad (R27)$$

$$y_1, y_2, y_3, y_4, y_5, y_6 \in \{0, 1\} \qquad (R28)$$

$$x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{23}, x_{24}, x_{25}, x_{26}, x_{34}, x_{35}, x_{36},$$
$$x_{45}, x_{46}, x_{56} \in \{0, 1\} \qquad (R29)$$

As restrições R(23-25) asseguram que ou um item original ou o seu duplicado é empacotado, seguindo a correspondência de índices apresentado na Tabela 5.
Considerando a restrição R24 do modelo acima:

$$y_3 + x_{13} + x_{23} + y_4 + x_{14} + x_{24} + x_{34} = 1 \qquad (R24)$$

De acordo com a Tabela 5, o índice 3 corresponde ao item B, e o índice 4 corresponde ao item duplicado de B com rotação de 90^{o}, isto é, Br. Consequentemente, ou i_3 (item com índice 3) inicializa o nível (y_3) ou i_3 é empacotado num nível inicializado por um item com um índice mais baixo (x_{13}, x_{23}), ou então o item duplicado de i_3, que neste caso é i_4 (de acordo com a Tabela 5), inicializa o nível (y_4) ou i_4 é empacotado num nível incializado por um item com um índice mais baixo (x_{14}, x_{24}, x_{34}).

2.3.3 Três estágios

Os modelos de empacotamento analisados anteriormente (2LBP e 2LBPr) são, tal como foi referido, modelos a dois estágios, em que o empacotamento de placas é processado em duas fases distintas: a primeira fase consiste em processar cortes horizontais de ponta a ponta (gerando níveis de corte), e a segunda fase consiste em processar cortes verticais de ponta a ponta, obtendo-se os itens.

Ao adicionar uma fase de corte ao problema de duas fases, é possível reduzir a área não utilizada dos objectos. Iremos analisar um modelo de corte e empacotamento para problemas 2BP de 3 estágios (2BP-3 restrito) introduzido por Puchinger e Raidl [10].

O processo de corte a três estágios realiza-se com cortes horizontais sucessivos no primeiro estágio, cortes verticais sucessivos no segundo estágio, e novamente cortes horizontais no terceiro e último estágio.

Uma solução válida para o modelo 2BP-3 restrito é constituída por um conjunto de placas finito, sendo cada placa desse conjunto dividida em níveis. Cada nível é dividido em diferentes pilhas, e cada pilha é preenchida por um conjunto de itens com largura idêntica. A ordem segundo a qual são inseridos os itens em cada pilha não afecta a validade da solução nem o seu valor objectivo.

Uma solução do modelo (2BP-3 restrito) poderá conter o máximo de:

- n pilhas - cada pilha é identificada pelo índice do maior item nela empacotado, isto é, o item com maior índice empacotado na pilha.
- n níveis - o identificador de cada nível corresponde ao identificador da pilha empacotada nesse nível com maior altura.
- n placas - o identificador de cada placa corresponde ao menor índice do conjunto de níveis contidas nessa placa.

O problema tem uma restrição adicional que é a pilha com maior altura empacotada num nível (que inicializa o nível e define o seu identificador) ser preenchida apenas com um único item. Consequentemente, o item mais alto (com o índice mais baixo) empacotado num nível, define a altura dessa nível. Por este motivo é que o modelo se intitula 2LBP-3 restrito.

Antes do modelo ser processado, o conjunto dos itens é ordenado por valores de altura não crescentes.

O modelo 2BP-3 restrito, é constituído pelas seguintes variáveis de decisão:

$x_{ji} \in \{0,1\},\ j = 1...n,\ i = j...n$

$$x_{ji} = \begin{cases} 1 & \text{se e só se o item } i \text{ está empacotado na pilha } j \\ 0 & \text{caso contrário} \end{cases}$$

$\beta_{kj} \in \{0,1\},\ k = 1...n,\ j = k...n$

$$\beta_{kj} = \begin{cases} 1 & \text{se e só se a pilha } j \text{ está empacotada no nível } k \\ 0 & \text{caso contrário} \end{cases}$$

$\gamma_{lk} \in \{0,1\},\ l = 1...n,\ k = 1...n$

$$\gamma_{lk} = \begin{cases} 1 & \text{se e só se o nível } k \text{ está empacotado na placa } l \\ 0 & \text{caso contrário} \end{cases}$$

O modelo de programação linear do modelo 2BP a três estágios restrito pode ser definido da seguinte forma:

$$Min \sum_{l=1}^{n} \gamma_{l,l} \qquad (10)$$

sujeito a:

$$\sum_{j=1}^{i} x_{ji} = 1 \qquad \forall i = (1...n) \qquad (11)$$

$$x_{ji} = 0 \qquad \forall j = (1...n-1),\ \forall i > j\ -\ w_i \neq w_j \vee h_i + h_j > H \qquad (12)$$

$$\sum_{k=1}^{j} \beta_{kj} = x_{jj} \qquad \forall j = (1...n) \qquad (13)$$

$$\sum_{i=j}^{n} h_i x_{ji} \leq \sum_{k=l}^{j} h_k \beta_{kj} \qquad \forall j = (1...n-1) \qquad (14)$$

$$\sum_{j=k}^{n} w_j \beta_{kj} \leq W \beta_{kk} \qquad \forall k = (1...n-1) \qquad (15)$$

$$\sum_{l=1}^{k} \gamma_{lk} = \beta_{kk} \qquad \forall k = (1...n) \tag{16}$$

$$\sum_{k=l}^{n} h_k \gamma_{lk} \leq H\gamma_{l,l} \qquad \forall l = (1...n-1) \tag{17}$$

$$x_{ji} \in \{0,1\} \qquad j = (1...n),\ i = (j...n) \tag{18}$$

$$\beta_{kj} \in \{0,1\} \qquad k = (1...n),\ j = (k...n) \tag{19}$$

$$\gamma_{lk} \in \{0,1\} \qquad l = (1...n),\ k = (l...n) \tag{20}$$

Função objectivo (10): A função objectivo minimiza o número de placas usado na solução do modelo. Se $\gamma_{l,l} = 1$, então a placa l está a ser usada.

Restrição (11): A primeira restrição (11) assegura que cada item i é empacotado exactamente uma vez.

Restrição (12): Esta restrição garante que os itens empacotados na mesma pilha têm a mesma largura, e que a altura somada de cada par de itens empacotados numa pilha não excede a altura total da placa.

Restrição (13): A restrição representada na expressão (13) assegura que cada pilha j a ser utilizada é empacotada exactamente uma vez no nível k (Se a pilha j contem o item j, então $x_{jj} = 1$).

Restrição (14): A restrição (14) assegura que a altura de cada pilha (calculada pela soma das alturas dos itens empacotados em cada pilha) nunca excede a altura da stripe k associada (que é idêntica à altura do item k, devido à restrição adicional introduzida pelos autores).

Restrições (13) e (14): Implicam que nenhum item é empacotado numa pilha não utilizada.

Restrição (15): Garante que a largura das placas (W) não deve ser excedida por nenhum

nível k. Esta restrição também impede o empacotamento de pilhas em níveis que não são utilizadas ($\beta_{kk} = 0$).

Restrição (16): Esta restrição assegura que cada nível k utilizado é empacotado em exactamente uma placa.

Restrição (17): Assegura que a altura de cada placa (H) não é excedida pela altura total dos níveis empacotados. Esta restrição também garante que não há níveis empacotados em placas que não são utilizados ($\gamma_{l,l} = 0$).

Restrições (18), (19), (20): As três últimas restrições do modelo definem que as variáveis x, β, γ são binárias, e também definem os seus domínios.

2.4 Métodos heurísticos

Na secção anterior, analisámos uma série de modelos exactos para resolver problemas de e empacotamento a duas dimensões. Teoricamente estes modelos são capazes de nos fornecer a solução óptima para qualquer problema. Porém, devido ao seu elevado grau de complexidade, na prática torna-se muitas vezes impossível obter a solução óptima em tempo útil de processamento.

Deste modo, sempre que a utilização de modelos exactos não for viável (isto é, caso não se obtenha uma solução em tempo de execução útil), torna-se necessário recorrer a uma estratégia alternativa a fim de se encontrar uma solução de qualidade satisfatória para o problema. Nestes casos, a utilização de heurísticas revela-se uma boa opção.

Os métodos heurísticos permitem gerar soluções (frequentemente próximas da solução óptima) de menor qualidade que os métodos exactos, mas de forma mais rápida. Nesta secção iremos analisar alguns métodos heurísticos adequados para a resolução de problemas 2BP.

Lodi et al. [8] apresentaram quatro heurísticas distintas, cada uma direccionada a tipos diferentes de problemas de corte e empacotamento a duas dimensões com objectos homogéneos (2BP). Os autores criaram uma classificação para distinguir os diferentes tipos de problemas, combinando as seguintes características:

- Orientação dos itens:
  - **O** - Itens orientados: Os itens não podem sofrer qualquer tipo de rotação.
  - **R** - Itens não orientados: Os itens podem sofrer uma rotação de 90^{o}.
- Tipo de corte:
  - **G** - Corte guilhotinável: Os cortes nos objectos são realizados de ponta a ponta, para se obterem os itens.
  - **F** - Corte livre: Os itens podem ser obtidos através de cortes ortogonais nos objectos, sem que seja necessário que os cortes sejam realizados de ponta a ponta. Isto é, admite-se qualquer tipo de padrão de corte, desde que as margens dos objectos não sejam violadas e os itens não se sobreponham.

Os tipos de problemas definidos pela classificação são:

- Itens orientados + Corte guilhotinável: (2BP - O - G)

- Itens não orientados + Corte guilhotinável: (2BP - R - G)
- Itens orientados + Corte livre: (2BP - O - F)
- Itens não orientados + Corte livre: (2BP - R - F)

Considerando os modelos exactos analisados anteriormente, podemos classificar os modelos 2LBP e 2BP-3e como sendo para o problema (2BP - O - G), e o modelo 2LBPr como sendo para o problema (2BP - R - G).
Iremos analisar as seguintes heurísticas apresentadas por Lodi et al. [8], sendo cada uma orientada a um dos tipos de problemas da classe anterior:

- KP_{OG} para problemas (2BP - O - G).
- KP_{RG} para problemas (2BP - R - G).
- AD_{OF} para problemas (2BP - O - F).
- TP_{RF} para problemas (2BP - R - F).

2.4.1 Heurística *KP* - Itens orientados, corte guilhotinável

A heurística KP_{OG} é para problemas (2BP - O - G). Trata-se de um algoritmo de corte e empacotamento a dois estágios, sendo que no primeiro estágio empacota itens por diferentes níveis (fase de empacotamento em rolos), e no segundo estágio empacota os níveis criados em placas. O processo de empacotamento dos itens em níveis, é determinado pela resolução de problemas da mochila (problemas *Knapsack*, de onde deriva o nome da heurística KP_{OG}).
Um problema da mochila pode ser definido da seguinte forma:

$KP01$

No **Problema da Mochila 0-1 (KP01)** são dados:

- um conjunto de itens, cada um com um lucro p_j e um peso c_j, $(j = 1...n)$;
- uma capacidade q.

Garantindo que o peso total dos itens seleccionados não excede a capacidade q, o objectivo do problema da mochila é seleccionar um subconjunto de itens, maximizando o lucro total dos itens seleccionados. .

Em cada iteração da primeira fase do algoritmo (fase de empacotamento em rolos),

- é inicializado um novo nível com o item mais alto por empacotar $\boxed{j^*}$;
- O processo de empacotamento por níveis é terminado através da resolução de uma instância do problema **KP01** descrito acima, sendo que para cada item j por empacotar, temos que:
  - lucro $\boxed{p_j = w_j h_j}$
  - peso $\boxed{c_j = w_j}$
  - capacidade $\boxed{q = W - w_j^*}$

O algoritmo KP_{OG} pode ser definido em pseudo-código da seguinte forma:

KP_{OG}

1. Os itens são ordenados por ordem não crescente de altura.

   (Fase 1:)

2. $\boxed{\textbf{repetir}}$
   (a) É criado um novo nível no rolo, empacotando-se o primeiro item por seleccionar.
   (b) Resolve-se a instância **KP01** (Problema da Mochila 0-1) associada ao nível criado.
   (c) Os itens seleccionados em **KP01** são empacotados.

   $\boxed{\textbf{enquanto} \text{ (não estiverem empacotados todos os itens)}}$

3. Sejam $\hbar_1, ..., \hbar_S$ os valores das alturas dos níveis resultantes

   (Fase 2:)

4. É determinada uma solução de 2BP através da resolução de uma instância de empacotamento a uma dimensão (1BP - *One-dimensional Bin Packing problem*), em que os itens correspondem aos S níveis tendo associados as alturas $\hbar_1, ..., \hbar_S$, e capacidade H.

2.4.2 Heurística *KP* - Itens não orientados, corte guilhotinável

Lodi et al. [8] apresentaram a heurística KP_{RG} de corte guilhotinável, semelhante à heurística KP_{OG}, com a diferença de possibilitar uma rotação de 90^{o} nos itens. À semelhança da estrutura da heurística KP_{OG}, a heurística KP_{RG} é constituída por duas fases distintas.

- Na primeira fase, os itens são empacotados por níveis em rolos através do problema da mochila 0-1, que é apresentado na secção anterior.
- Na segunda fase, os rolo empacotado na primeira fase é particionado em diferentes placas.

Na heurística KP_{RG}, os itens são inicialmente ordenados por ordem não crescente da dimensão mais curta (altura ou largura) sendo depois orientados horizontalmente, ou seja, com o lado maior na horizontal. A orientação horizontal é sempre usada na inicialização de níveis de empacotamento permitindo melhores resultados na primeira fase de empacotamento.

Para cada nível (com altura h^*), a instância do problema da mochila 0-1 inclui todos os itens por empacotar. Os itens são preferencialmente orientados na vertical, caso as suas dimensões verticais não ultrapassem h^*, ou na horizontal caso contrário.

Assim que é encontrada uma solução com um número finito de placas através da heurística KP_{OG}, repete-se o seu processamento para calcular uma solução alternativa através de uma nova instância. Na nova instância é criado um pseudo-item para cada nível gerado na primeira solução com as seguintes dimensões:

1. altura do nível;
2. espaço horizontal utilizado no nível.

Os pseudo-itens cuja arestas mais longas não excedem a altura da placa (H) orientam-se verticalmente e é executada a heurística KP_{OG} para a nova instância. No final da execução é seleccionada a solução que apresenta melhor qualidade.

A heurística KP_{RG} pode ser definida da seguinte forma:

KP_{RG}

1. Os itens são ordenados por ordem não crescente de valores $min\{w_j, h_j\}$, e horizontalmente orientados.
2. Executa-se a heurística KP_{OG}.
3. Seja z_1 a solução obtida pela instância KP_{OG}.
4. Definem-se s pseudo-itens com dimensões ϖ_i, $\hbar_i$ $(i = 1, ..., s)$, sendo s o número de níveis de empacotamento de z_1.
5. Os pseudo-itens são orientados verticalmente i tal que $max\{\varpi_i, \hbar_i\} \leq H$.
6. Executa-se o algoritmo KP_{OG} para a respectiva instância (2BP — O — G) construida com base nos pseudo-itens, considerando-se a solução resultante z_2.
7. Definir como solução final do problema KP_{RG} como $z = min\{z_1, z_2\}$.

2.4.3 Heurística das Direcções Alternadas - Itens orientados, corte livre

Para problemas 2BP onde os itens não sofrem rotação e as restrições de corte guilhotinável não se aplicam (não sendo realizado o empacotamento dos itens por níveis), Lodi et al. [8] introduziram uma heurística chamada Direcções Alternadas (*Alternate Directions*).

A heurística AD_{OF} que explora a viabilidade de padrões de itens não guilhotináveis (Corte livre, ou *Free Cutting*), empacotando os itens em placas sequencialmente, da **esquerda para a direita** e da **direita para a esquerda** de forma alternada, na posição mais baixa possível.

O algoritmo começa por ordenar os itens por valores de altura não crescentes, e calcula um valor para o limite inferior L para a solução óptima.

Podemos considerar o valor contínuo L_0 como limite inferior, calculado a partir da expressão:

$$L_0 = \sum_{j=1}^{n} \frac{w_j h_j}{WH}$$

Após o cálculo de L_0, são inicializadas L_0 placas e um subconjunto de itens é empacotado na base destes, seguindo a estratégia *best-fit decreasing*. Os itens restantes são empacotados em diferentes bandas, de acordo com a direcção considerada no momento (isto é, ou esquerda

para a direita, ou direita para a esquerda) em relação à placa. Se a direcção for **esquerda para a direita** (ou respectivamente **direita para a esquerda**):

1. O primeiro item da banda é empacotado com o lado **esquerdo** (resp. direito) junto à extremidade **esquerda** (resp. direita) da placa, na posição mais abaixo possível;

2. Cada item subsequente é empacotado com o lado **esquerdo** (resp. direito) junto ao lado **direito** (resp. esquerdo) do **item anterior na banda**, na posição mais abaixo possível.

É de salientar o facto de os itens empacotados (da **esquerda para a direita**) na fase de inicialização constituirem a primeira banda de cada placa. Depois deste passo, a direcção considerada passa a ser da **direita para a esquerda**.
Na parte iterativa do algoritmo, para cada placa, são pesquisados todos os itens não empacotados, havendo a possibilidade de empacotar todos os itens na direcção considerada.
Sempre que se esgota o conjunto de itens por empacotar, de tamanho compatível com o espaço disponível numa determinada banda, a direcção de empacotamento é alterada.
Quando não é possível empacotar qualquer item, quer na direcção **esquerda-direita** quer na direcção **direita-esquerda**, o algoritmo avança para a placa seguinte (caso exista), ou inicializa uma nova placa.
Uma vantagem associada à estratégia de alternação da direcção de empacotamento dos itens, é o maior aproveitamento do espaço residual gerado na direcção usada numa iteração anterior. Considerando a iteração inicial da heurística, uma vez que os itens são ordenados por ordem não crescente antes da fase de empacotamento, é comum que o desperdício seja gerado diagonalmente. Ou seja, há mais área preenchida verticalmente à direita do que à esquerda. Na iteração seguinte, o item por empacotar mais alto é alocado à direita, aproveitando a zona inferior da placa com maior área livre.

O algoritmo das Direcções Alternadas pode ser definido da seguinte forma:

AD_{OF}

1. Os itens são ordenados por valores de h_j não crescentes.
   (Fase 1)
2. É calculado um limite inferior para a solução do problema L, e inicializam-se L placas.
3. **para** $(j := 1)$ **até** (n)
   executar
   (a) **se** (item j pode ser empacotado na base de uma placa existente)
   então empacota-se j, alinhado à esquerda, na base da placa em que o espaço horizontal residual é mínimo.

   (Fase 2)
4. Registar i:=0 (i é o índice da placa a ser utilizada. Quando i é incrementado, é inicalizada uma nova placa).
5. **repetir**
   (a) $i := i + 1$.
   (b) **direita-esquerda** = $true$.
   (c) $n\_falhas := 0$.
   (d) **repetir**
      i. Seja j o primeiro item por empacotar que pode ser empacotado na placa i, de acordo com o valor actual da *flag* **direita-esquerda**.
      ii. **se** $(j == Nil)$
      então
         A. $n\_falhas := n\_falhas + 1$
         B. **direita-esquerda** := *not* **direita-esquerda**
      caso contrário
         A. j empacotado na placa i de acordo com **direita-esquerda**.
         B. $n\_falhas := 0$.
   até $n\_falhas == 2$.

   até todos os itens estarem empacotados.

2.4.4 Heurística do Perímetro em Contacto - Itens não orientados, corte livre

A última heurística apresentada por Lodi et al. [8], é relativa a problemas de corte livre com itens não orientados, isto é, itens que podem sofrer uma rotação de 90º (Problemas do tipo (2BP - R - F)). A heurística tem o nome original de *Touching Perimeter*, que pode ser traduzido como Perímetro em Contacto, a que nos referiremos como TP_{RF}.

O algoritmo começa por ordenar os itens por valores de área não crescente (os empates entre itens com área similar são resolvidos, ordenando os itens por valores não crescentes de $min\ \{w_j, h_j\}$, orientando-os depois horizontalmente).

É calculado um limite inferior L para a solução do problema, e são inicializadas L placas. Este limite inferior pode ser calculado da mesma forma que na heurística das Direcções Alternadas:

$$L_0 = \sum_{j=1}^{n} \frac{w_j h_j}{WH}$$
$$(L = L_0)$$

O algoritmo empacota um item de cada vez, ou numa placa existente, ou numa nova placa inicializada. O primeiro item a ser empacotado numa placa é sempre colocado no canto inferior esquerdo.

Cada item subsequente é empacotado na posição normal, ou seja:

- com a aresta inferior em contacto com a base da placa ou com a aresta superior de outro item;
- e com a sua aresta esquerda em contacto ou com a extremidade esquerda da placa, ou com a aresta direita de outro item.

A escolha da placa e da posição de empacotamento é realizada através do cálculo de uma pontuação. A pontuação de um determinado item corresponde à percentagem do seu perímetro em contacto com as extremidades da placa ou com as arestas de outros itens empacotados.

O algoritmo favorece a escolha de padrões onde os itens empacotados não deixam pequenas áreas residuais que dificultem os futuros empacotamentos.

Para cada posição de empacotamento candidata, a pontuação de um item é calculada duas vezes, para as duas orientações possíveis do item (isto é, vertical ou horizontal) caso as duas sejam viáveis, e é seleccionado o valor da pontuação mais alta. Os casos de empate de pontuações são resolvidos através da escolha da placa com área empacotada máxima. Para alternativas equivalentes no posicionamento de itens, é dada prioridade ao posicionamento inferior e à esquerda.

A heurística do Perímetro em Contacto pode ser definida em pseudo-código da seguinte forma:

TP_{RF}

1. Os itens são ordenados por valores não crescentes de $w_j h_j$, e são depois orientados horizontalmente, ou seja, o valor da largura de cada item corresponde à dimensão da aresta maior.
   (Fase 1)
2. É calculado um limite inferior L para a solução do problema e são inicializadas L placas.
   (Fase 2)
3. **para** ($j := 1$) **até** (n)
   início
   (a) $pontos := 0$;
   (b) **Para cada** (Posição normal de empacotamento numa placa inicializada)
   início
   i. Seja $pontos1$ e $pontos2$ as pontuações associadas às duas orientações possíveis (horizontal ou vertical);
   ii. $pontos := max\{pontos, pontos1, pontos2\}$
   fim
   (c) **se** ($pontos > 0$)
   então (empacotar item j na placa, com a posição e orientação correspondentes a $pontos$).
   caso contrário inicializar nova placa, e empacotar horizontalmente o item j.
   fim

Consideramos a instância da Tabela 1 para construir um exemplo de aplicação da heurística *Touching Perimeter*. Definimos os objectos com as dimensões de largura $W = 10$ e altura $H = 8$. Os itens são ordenados por valores não crescentes de $w_i h_i$, e depois orientados horizontalmente. O valor calculado para L_0 é $\lceil 1.2375 \rceil = 2$, sendo por isso inicializadas 2 placas antes do empacotamento dos itens.
Os valores apresentados na Tabela 6 exemplificam a execução do algoritmo, e apresentam as pontuações calculadas sucessivamente durante o empacotamento. Para cada item, a pontuação mais alta determina em que placa é empacotado e com que direcção, sendo seleccionada a maior das seguintes pontuações:

- Pontuação para a placa 1, com orientação horizontal.
- Pontuação para a placa 1, com orientação vertical.
- Pontuação para a placa 2, com orientação horizontal.
- Pontuação para a placa 2, com orientação vertical.

Se a pontuação for X, então o item não pode ser empacotado com a orientação e na placa considerados nessa iteração do algoritmo. Isto é, se o item i tiver pontuação X para orientação horizontal na placa q, então o item i não pode ser empacotado horizontalmente na placa q.
Os itens são ordenados por valores não crescentes de $w_i h_i$, sendo gerada a seguinte sequência de itens:

$$A, D, B, C, F, G, H, E$$

O processamento da heurística do Perímetro em Contacto com base na sequência anterior é realizado da seguinte forma:

1. O item A, por ser o primeiro item a ser empacotado, tem igual pontuação nas duas dimensões, e nas duas placas disponíveis. É considerada a pontuação calculada em primeiro lugar: orientação horizontal na placa 1, na posição (0,0).
2. O item D tem maior perímetro em contacto, se for empacotado na placa 2 verticalmente na posição (0,0), ficando com três arestas em aderência com os limites da área empacotada.

3. Sendo o item B um quadrado, é necessário apenas apurar em que placa e posição é empacotado, uma vez que a sua orientação é equivalente. A percentagem de perímetro em contacto é de 75% na placa 1 e 50% na placa 2, ficando o item alocado na placa 1. Há neste caso, dois posicionamentos possíveis a gerarem a mesma pontuação: A posição (0,4) e a posição (6,0). Uma vez que o posicionamento inferior tem primazia sobre o posicionamento à esquerda, é escolhida a posição (6,0).

4. Das quatro alternativas de empacotamento para o item C, três geram um perímetro em contacto de 50% com duas arestas encostadas. Apenas o posicionamento na placa 1 com orientação vertical, garante o contacto de três arestas do item, na posição (0,4) em cima do item A.

5. No empacotamento do item F, o posicionamento vertical na placa 1 é impossível, por serem ultrapassados os limites de empacotamento. As restantes opções são equivalentes, com uma percentagem do perímetro do item em contacto de 50%. Uma vez mais é seleccionada a primeira alternativa calculada, que é o empacotamento na placa 1 com orientação horizontal na posição (3,4).

6. O item G, tal como o item B, é um quadrado. A placa em que o empacotamento do item gera maior pontuação é a placa 1 na posição (3,6), com três arestas encostadas ao tecto do item por cima, ao item C à esquerda e ao item F em baixo.

7. O empacotamento do item H tem características similares ao do item C: três alternativas de empacotamento iguais de 50%, e uma superior de 80%, com orientação vertical na placa 1 na posição (9,4).

8. O último item a ser empacotado é o item E. Na placa 1, a orientação vertical não é possível, pelo que a pontuação associada a esta alternativa é X. As restantes alternativas de empacotamento têm pontuação equivalente, garantindo duas arestas em contacto total. É escolhida a primeira alternativa calculada, ou seja, na placa 1 com orientação horizontal, na posição (5,6).

Na Figura 6 é possível observar um esquema representativo do resultado de execução da instância da Tabela 1 pela heurística *TP*.

Tabela 6: Cálculo dos *scores* para a instância da Tabela 1 na heurística *TP*, com placas de tamanho $H = 8$ e $W = 10$.

Indíce i	cod_i	w_i	h_i	$w_i h_i$	Perímetro	$pontos_{horiz}^{placa1}$	$pontos_{vert}^{placa1}$	$pontos_{horiz}^{placa2}$	$pontos_{vert}^{placa2}$
1	A	6	4	24	20	50%	50%	50%	50%
4	D	8	3	24	22	41%	45%	50%	64%
2	B	4	4	16	16	75%	75%	50%	50%
3	C	4	3	12	14	50%	71%	50%	50%
6	F	6	2	12	16	50%	X	50%	50%
7	G	2	2	4	8	75%	75%	50%	50%
8	H	4	1	4	10	50%	80%	50%	50%
5	E	3	1	3	8	50%	X	50%	50%

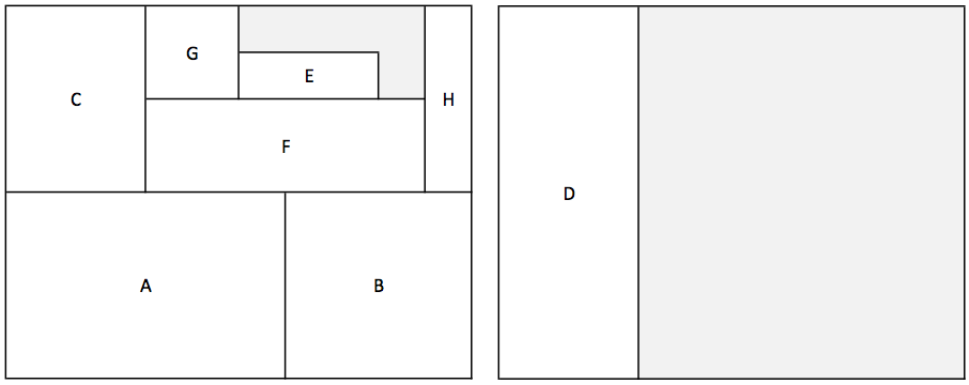

Figura 6: Esquema do resultado de execução da instância da Tabela 1 pela heurística *TP*.

2.4.5 Heurística *HBP* - Itens orientados/não orientados, corte livre

Nesta secção analisamos a heurística HBP, apresentada por Boschetti e Mingozzi [1] , para resolver problemas 2BP com corte livre, em que os itens podem ou não sofrer uma rotação de 90º (Problemas do tipo (2BP - O - F) e (2BP - R - F), segundo a tipologia de Lodi et al [8]). O problema 2BP considerado aqui pode ser formulado da seguinte forma.

Dado um problema de empacotamento em placas de iguais dimensões, assumimos que cada item $j \in J$ tem um parâmetro ρ_j associado, tal que:

$$\rho_j = \begin{cases} 1 & \text{se e só se o item } j \text{ tiver uma rotação de } 90^{\text{o}} \\ 0 & \text{caso contrário} \end{cases}$$

É considerada a área de cada item j ($j \in J$) $\boxed{a_j = w_j \times h_j}$.

Cada placa é localizada no quadrante positivo do sistema de coordenadas com a sua **origem** (isto é, o canto inferior esquerdo) localizada na posição (1,1), e com a suas arestas inferior e esquerda paralelas aos eixos dos xx e dos yy, respectivamente.

Esta heurística ordena os itens por um determinado critério, e tenta alocar os itens considerando uma placa de cada vez. Quando mais nenhum item consegue ser alocado na placa actual, fecha-se a placa e abre-se uma nova. O processo termina quando todos os itens ficam empacotados. Este procedimento é então repetido trocando-se a ordem dos itens. O algoritmo pára quando o número de placas usado iguala um limite mínimo para a solução do problema, ou quando um número de iterações máximo é alcançado.

A solução para o problema é representada especificando para cada item $j \in J$

- a placa com index π_j onde j se localiza;
- as coordenadas (p_j, q_j) do seu canto inferior esquerdo (origem do item);
- o parâmetro de rotação ρ_j.

A heurística *HBP* tem como parte principal o algoritmo *HSolve*, que pode ser definido da seguinte forma:

Algoritmo *HSolve*

Seja n a cardinalidade de elementos do conjunto de itens J.

- **1 - Inicialização**

  1. Inicializa-se o parâmetro de preço ν_j para cada item $j \in J$, sendo J o conjunto de itens.
  2. Inicializa-se a melhor solução alocando cada item $j \in J$ numa placa diferente, tal que $\pi_j = j$, na posição $(p_j, q_j) = (1, 1)$, com rotação $r_j = 0$.
  3. Define-se $z^* = n$ e $iter = 1$. (*iter* significa a iteração actual)

- **2 - Optimização da solução actual**

  1. Define-se um novo conjunto de itens $J^{'} = J$.
  2. Seja $i = 1$ o índice da placa vazia a ser utilizada.
  3. Define-se $\pi_j^{'} = n, \forall j \in J$. Ou seja, para cada item, é atribuída uma placa com o índice n.
  4. Os itens do conjunto $J^{'}$ são ordenados por ordem não crescente do parâmetro de preço ν (Para $J^{'} = (j_1, j_2, ..., j_n)$ *temos* $\nu_{j1} \geq \nu_{j2} \geq ... \geq \nu_{jn}$).

- **3 - Preenchimento da placa** i

  1. Pesquisar para cada item $j \in J^{'}$ a melhor posição $(p_j^{'}, q_j^{'})$ e a melhor orientação $r_j^{'}$ para empacotar j na placa actual i. Se existir uma posição admissível, então o item j é adicionado à solução, definindo $\pi_j^{'} = i$ e $J^{'} = J^{'} \backslash \{j\}$. O método para calcular a melhor posição e orientação na alocação de itens é explicada mais adiante nas **Regras de alocação de itens** e **Regras do parâmetro de preço**.
  2. O passo 3 termina quando mais nenum item de $J^{'}$ pode ser empacotado na placa i.

- **4 - Verificar se todos os itens foram alocados**

  1. Seja $z = i$ o custo da solução resultante. Ou seja, z representa o número de placas usadas na solução.
  2. Se $J^{'} = \emptyset$, então avança-se para o **passo 5**.
  3. Se $J^{'} \neq \emptyset$ e $i < z^* - 1$, então é aberto uma nova placa, definindo $i = i + 1$ e retorna-se ao *passo 3*, ou caso contrário avança-se para o **passo 7**.

(Continuação)

- **5 - Actualizar a melhor solução**

  1. Caso z tenha melhor qualidade do que z^*, actualiza-se o valor de $z^* = z$, $\pi_j = \pi_j^{'}$, $(p_j, q_j) = (p_j^{'}, q_j^{'})$ e $r_j = r_j^{'}$, $\forall j \in J$.

- **6 - Verificar valor óptimo**

  1. Se z^* for igual a um limite inferior conhecido, então parar execução de HSolve.

- **7 - Actualizar parâmetro de preço dos itens**

  1. Se *iter* atingir um número máximo de iterações previamente estipulado, então parar execução de *HSolve*. Caso contrário, para cada item $j \in J$, actualizar o preço ν_j de acordo com a solução actual, incrementar *iter*, e recuar ao **passo 2**.

O método *HBP* pode ser executado repetidas vezes, modificando as regras de **preço** usadas nos **passos 1 e 7**, e modificando o **método de alocação de itens** usado no **passo 3**.

Uma versão melhorada do método *HBP* poderá ser alcançado através da execução do algoritmo *HSolve* conjugando todas as combinações possíveis dos diferentes métodos de alocação e das regras de preço.

O algoritmo *HBP* apresentado por Boschetti e Mingozzi [1] é definido da seguinte forma:

Algoritmo *HBP*

1. Definir $U_{BM} = \infty.$, ou seja, a solução inicial é o pior caso possível, que corresponde a atribuir uma placa a cada item existente, correspondendo ao máximo número de placas possível.
2. Para cada **método de alocação de itens** AC_1 e AC_2, e para cada **regra de preço** PU_1 e PU_2, executar **passo 3**. Os métodos AC_1 e AC_2, bem como as regras PU_1 e PU_2 são explicadas mais adiante, respectivamente nas **Regras de alocação de itens** e **Regras do parâmetro de preço**.
3. Executar **algoritmo *HSolve*** usando as regras seleccionadas e actualizar limite máximo à solução final $U_{BM} = min\ \{z^*, U_{BM}\}$

Regras de alocação de itens O critério geral usado no algoritmo *HSolve* para alocar um item j numa placa, consiste em encontrar a posição onde j não se sobreponha aos itens já alocados, de forma a que não seja possível movê-lo para a esquerda e/ou para baixo, já que a arestas esquerda e inferior de j são adjacentes às arestas dos itens empacotados ou da placa.

Cada posição admissível para o item j é representada pelo triplo (p, q, r), onde r representa a rotação do item, e o par (p, q) representa as coordenadas da origem do item.

Seja $F(j, B_i)$ o conjunto de todos os triplos admissíveis (que satisfazem o critério descrito acima), para alocar o item j na placa i, onde $B_i = \{j \in J \ : \pi_j^{'} = i\}$ é o subconjunto de itens já alocados na placa.

Se $|F(j, B_i)| = 1$, então o item j é alocado na posição $(p, q, r) \in F(j, B_i)$, considerando $p_j^{'} = p$, $q_j^{'} = q$ e $r_j^{'} = r$. Se $|F(j, B_i)| > 1$, então a melhor posição é escolhida pelo critério de empacotamento, em que o *layout* da placa mais compatível para o item j é obtido empacotando o item j na posição com menos área por ocupar à esquerda e em baixo.

Seja $f^* = min\ \{f(p, q, r) : (p, q, r) \in F(j, B_i)\}$ e $F^{'}(j, B_i) = \{(p, q, r) \in F(j, B_i) : f(p, q, r) = f^*\}$. O algoritmo *HBP* usa as duas seguintes regras AC_1 e AC_2, para alocar o item j no **passo 3** do algoritmo *HSolve*:

AC1 Empacotar j na posição $(p_j^{'}, q_j^{'}, r_j^{'})$ definindo sequencialmente $p_j^{'}$, $q_j^{'}$ e $r_j^{'}$ da seguinte forma: $p_j^{'} = min\{p : (p, q, r) \in F^{'}(j, B_i)\}$, $q_j^{'} = min\{q : (p_j^{'}, q, r) \in F^{'}(j, B_i)\}$ e $r_j^{'} = min\{r : (p_j^{'}, q_j^{'}, r) \in F^{'}(j, B_i)\}$.
AC2 Empacotar j na posição $(p_j^{'}, q_j^{'}, r_j^{'})$ definindo sequencialmente $q_j^{'}$, $p_j^{'}$, e $r_j^{'}$ da seguinte forma: $q_j^{'} = min\{q : (p, q, r) \in F^{'}(j, B_i)\}$, $p_j^{'} = min\{p : (p, q_j^{'}, r) \in F^{'}(j, B_i)\}$ e $r_j^{'} = min\{r : (p_j^{'}, q_j^{'}, r) \in F^{'}(j, B_i)\}$.

Regras do parâmetro de preço A cada iteração do algoritmo *HBP*, são usadas diferentes funções de cálculo do parâmetro de preço $\nu_j,\ j \in J$. O objectivo consiste em atribuir preços altos a itens com maiores dificuldades no processo de empacotamento. São consideradas quatro diferentes regras para definir um preço inicial ν_j:

- IP_1**:** A área, $\nu_j = a_j$;
- IP_2**:** A largura, $\nu_j = w_j$;
- IP_3**:** A altura, $\nu_j = h_j$;
- IP_4**:** O perímetro, $\nu_j = 2w_j + 2h_j$.

No final de cada iteração do algoritmo *HSolve*, é actualizado o preço $\nu_j, j \in J$, com base na última solução com custo z. Para cada item j localizado na placa $\pi_j^{'} > \frac{1}{2}z$, o preço é aumentado, enquanto que para cada item j localizado na placa $\pi_j^{'} \leq \frac{1}{2}z$, o preço é diminuido. Os preços são actualizados da seguinte forma: $\nu_j = \alpha\nu_j$, se $\pi_j^{'} \leq \frac{1}{2}z$ e $\nu_j = \beta\nu_j$, se $\pi_j^{'} > \frac{1}{2}z$, onde $\alpha < 1$ e $\beta > 1$. Os valores de α e β podem ser definidos de forma estática (PU_1) ou aleatória (PU_2).

2.4.6 Heurística Chão-tecto

A heurística Chão-tecto (*Floor-Ceiling* - FC_{RG}) foi apresentada por Lodi et al. [8], e é direccionada a problemas (2BP - R - G). Os itens podem sofrer uma rotação de $90^{\underline{o}}$, e o corte é guilhotinável a duas fases. O algoritmo organiza os itens inicialmente por ordem não crescente da sua dimensão mais curta, e orienta-os horizontalmente.
Os algoritmos de empacotamento por níveis descritos até agora empacotam os itens sucessivamente por níveis, alinhando-os à esquerda, com a sua base em contacto com a base do nível. A particularidade do algoritmo Chão-tecto está em empacotar adicionalmente os itens alinhando-os à direita, com a aresta superior em contacto com o a linha superior do nível.

Sempre que um item é empacotado,

1. É equacionado o empacotamento com rotação;
2. As restrições de corte guilhotinável são verificadas, sendo que a posição de empacotamento do item poderá ser alterada em caso de violação.

Na segunda fase, os níveis resultantes da primeira fase são empacotados em diferentes placas, através da resolução de um problema 1BP.
O algoritmo FC_{RG} pode sofrer diversas alterações para ser adaptado a outros tipos de problemas de corte e empacotamento:

- FC_{RF}: Para modificar a característica de corte guilhotinável para corte livre, é necessário saltar o **passo 1** do algoritmo.

- FC_{OG}: Para retirar a característica de rotação dos itens, é necessário ordenar inicialmente os itens por ordem não crescente de altura, e saltar o **passo 1** do algoritmo.

- FC_{OR}: Para tirar a rotação e implementar o corte livre, é necessário ordenar os itens inicialmente por ordem não crescente de altura, e saltar os **passos 1 e 2** do algoritmo.

3 Descrição do problema

3.1 O caso de estudo *MPinto*

O desenvolvimento do algoritmo descrito nesta dissertação foi promovido em primeira instância por uma empresa de aplicação de revestimentos de superfícies da zona de Braga, a *MPinto*, com o objectivo de optimizar o seu processo de planeamento de corte de matéria-prima.

A instalação de revestimentos de chão e paredes preconizada pela *MPinto* é um processo industrial de corte a duas dimensões. A matéria-prima utilizada, na sua maioria linóleo e vinílleo, é providenciada por diversos fornecedores e comprada em rolos de diferentes tamanhos consoante o modelo e a marca do material.

O processo de preparação de obra é realizado nas instalações da empresa por uma equipa de colaboradores não inferior a dois elementos. A preparação da obra consiste num trabalho de estereotomia, ciência que estuda o corte regular de materiais de indústria e de construção.

Caso os projectos de arquitectura estejam disponíveis em ficheiros CAD, a equipa de preparação de obra utiliza ferramentas de cálculo de áreas geométricas disponibilizadas no *software* CAD utilizado na empresa para determinar as secções de corte a implementar no local de obra. Na ausência de ficheiros de *software* CAD com as plantas do projecto, as secções de corte são determinadas manualmente, quer a partir dos projectos de arquitectura em papel, quer através de um levantamento das dimensões das áreas de corte pretendidas.

O método de trabalho levado a cabo pela equipa de planeamento na definição das áreas de corte é pouco eficiente, moroso e envolve um grau elevado de intuição por parte da equipa de planeamento, contrapondo-se a uma metodologia sistemática que garanta a objectividade e coerência à replicação dos resultados do planeamento. Consequentemente, a planificação de obra varia inevitavelmente de colaborador para colaborador e não é garantida qualquer constância dos níveis de eficiência do processo produtivo, quer do ponto de vista do consumo de matéria-prima, quer do ponto de vista da produtividade dos próprios colaboradores.

Acresce a este facto o problema de ocasionalmente se gerarem pontos de discordância entre a *MPinto* e o arquitecto da obra ou outro agente decisor, pelo desfasamento observado entre

a visão que este ambiciona para o resultado final e o plano resultante da preparação que a equipa da *MPinto* produz. Os motivos da discordância baseiam-se quase sempre em critérios de preferência estética subjectivos. O presente trabalho não tem como objectivo solucionar esta questão, e portanto assumiremos que o plano resultante da preparação é aceite como tal pelos agentes de decisão envolvidos no projecto.

No final do processo de preparação, após a validação dos resultados, é entregue o plano à equipa da *MPinto* que procede à aplicação do material no local de obra. O plano de preparação é construído numa folha de cálculo em formato digital '.xls', constituído por duas secções distintas:

1. Entrada
2. Saída

Podemos observar um esquema representativo da estrutura do documento de preparação de obra da *MPinto* na Figura 7.

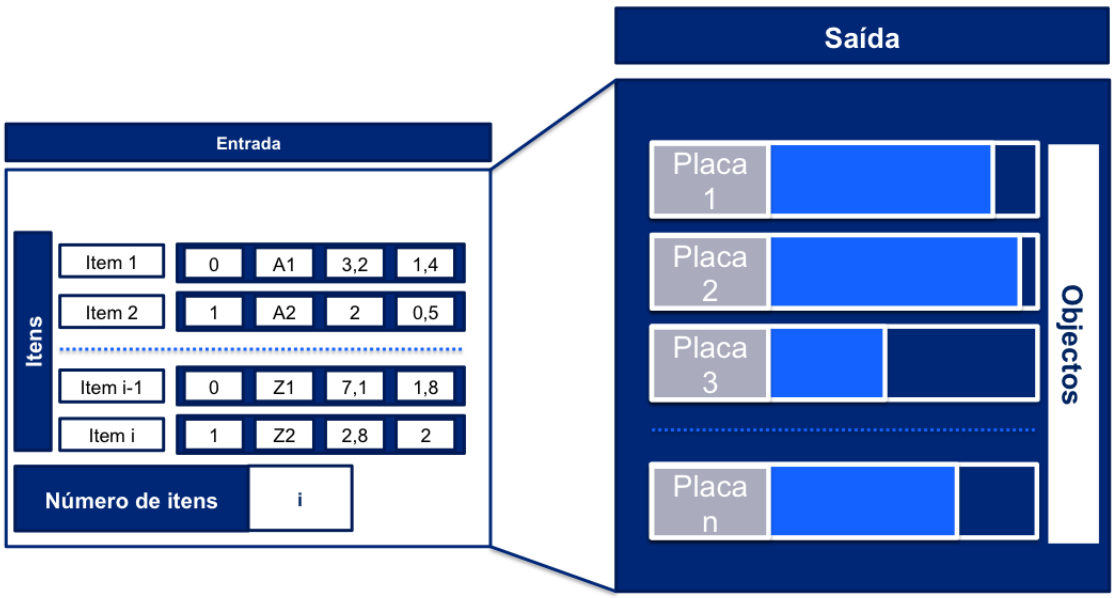

Figura 7: Estrutura do documento de preparação da *MPinto*.

A entrada é, como foi referido anteriormente na descrição de problemas de corte a duas dimensões, o conjunto de itens que correspondem às secções geométricas a serem cortadas. A

saída é respectivamente o conjunto de placas que correspondem aos rolos de matéria-prima.

Após a determinação das secções de corte no ficheiro CAD (ou manualmente através do planta em papel ou dos dados levantados, conforme o caso), estas são transferidas para a primeira secção do documento, e são representadas através dos seguintes atributos:

- **Piso** – Código numérico identificador do piso no projecto de arquitectura onde a secção de corte será instalada.
- **Código** – Código alfanumérico identificador da secção de corte.
- **Altura** – Altura da secção de corte representada no formato: metros, centímetros.
- **Largura** – Largura da secção de corte representada no formato: metros, centímetros.

Finalizada a identificação do conjunto de itens bem como o comprimento dos rolos de matéria-prima (placas), dá-se início ao processo de alocação de itens a placas, que constitui o problema de corte a duas dimensões estudado no presente trabalho. Na Figura 8 é possível observar um esquema que representa o processo de alocação de itens a placas com base na informação recolhida no documento de preparação.

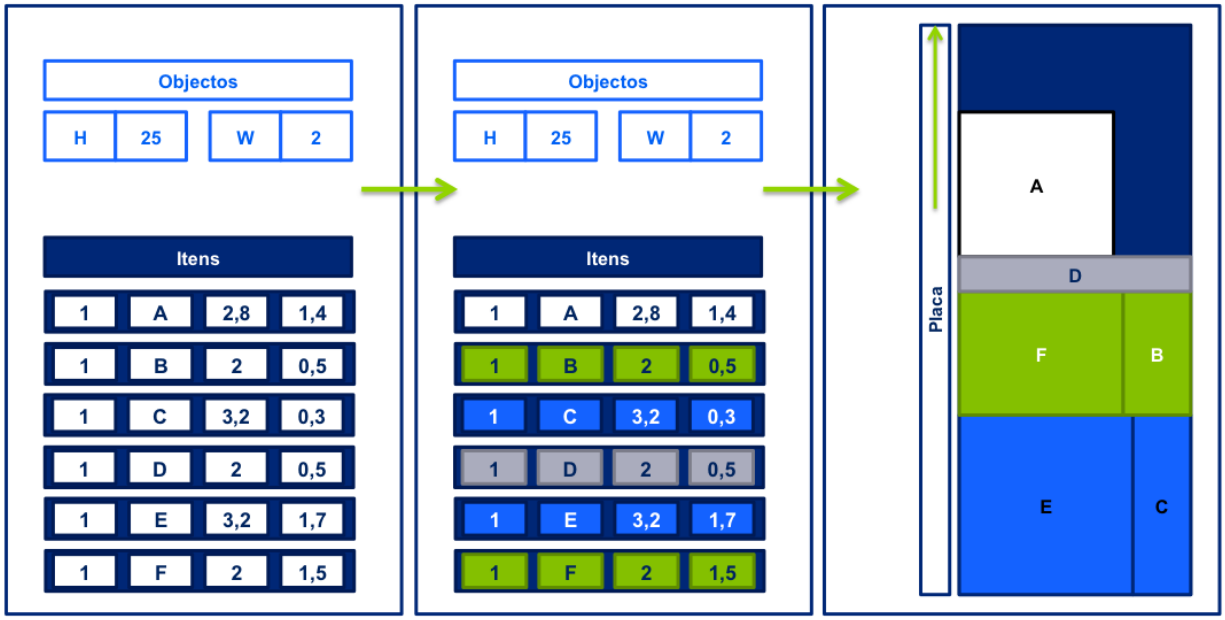

Figura 8: Processo de alocação de itens da *MPinto*.

A *MPinto* providenciou desde logo na fase inicial do projecto de investigação, algumas instâncias reais do seu processo de planeamento de obra. As soluções produzidas pela equipa de preparação apresentaram resultados de corte a duas dimensões em alguns casos de corte livre, e em outros casos guilhotinável até três fases, com rotação de itens.

Para chegar às soluções analisadas, como já foi referido anteriormente, não é utilizada uma metodologia sistemática para o processo de alocação de itens em objectos, embora se tenham identificado alguns pressupostos de planeamento, assumidos como dado adquirido pela equipa. Estes pressupostos são linhas orientadoras no processo de planeamento manual que não são seguidos de forma sistematizada durante o planeamento, concedendo espaço à arbitrariedade e intuição do colaborador:

- São alocados a montante os itens com largura equivalente à largura da placa.
- Os itens com largura inferior à largura da placa são agrupados em subconjuntos, tal que a largura máxima de cada subconjunto se aproxime tanto quanto possível da largura da placa, sem a exceder.
- Procura-se minimizar a área livre gerada na secção rectangular da placa ocupada por cada subconjunto de itens.

A segunda secção do documento contém a representação dos planos de corte dos rolos de matéria-prima. Tendo o conhecimento do comprimento fixo das placas, são alocados progressivamente os subconjuntos de itens gerados nos rolos abertos de forma manual e arbitrária, atribuindo-se maior prioridade aos rolos com menor dimensão (reciclagem de sobras), e aos rolos com maiores níveis de preenchimento. Sempre que a capacidade de alocação de um rolo de matéria-prima é esgotada, é aberto um novo rolo.

O resultado final da segunda secção do documento de planeamento resulta num conjunto finito de planos de corte de objectos que determinam a quantidade de rolos de matéria-prima a serem utilizados na fase de implementação do material de revestimento de superfícies no local de obra.

Verificando-se a ausência de material em *stock* definido pelo planeamento, impõe-se a necessidade de encomendar os rolos de matéria-prima em falta. Seja x o número de rolos abertos

durante o planeamento, e inexistentes em *stock*. A *MPinto* tem a possibilidade de comprar $(x-1)$ rolos inteiros, mais um rolo adicional com tamanho parcial, menor ao comprimento máximo disponível, com um custo por metro associado. O rolo de tamanho parcial corresponde ao rolo no planeamento com menor preenchimento, cuja altura preenchida se pretende minimizar a fim de se optimizar a matéria-prima consumida bem como os gastos financeiros associados.

3.2 Problema de empacotamento a duas dimensões com minimização da altura da placa menos preenchida em altura

Podemos considerar as soluções iniciais para alocação de superfícies de revestimento da *MPinto* como soluções de problemas de corte e empacotamento por níveis a três fases, com rotação de itens, e em alguns casos de corte livre com rotação de itens, considerando as soluções de algumas instâncias reais fornecidas pela empresa. A classificação do método é caracterizada pela solução atingida, independentemente de a equipa de estereotomia seguir uma metodologia sistemática ou não.

O objectivo de construir um modelo de corte e empacotamento para o caso *MPinto*, não consiste em modelar e replicar o método de alocação em vigor, mas sim construir um modelo capaz de fornecer soluções com qualidade satisfatória e reduzir drasticamente o seu tempo de execução.

Foi realizado um levantamento de modelos de empacotamento existentes que poderiam integrar uma ferramenta de planeamento do corte. Um desses casos foi o modelo de Lodi et al. [7], de empacotamento a duas dimensões por níveis a dois estágios, com e sem rotação de itens, que foi analisado na secção de revisão bibliográfica (Capítulo 2). A maior vantagem que este modelo tem associada, é o facto de possibilitar a determinação da solução óptima. Porém, para instâncias de maior dimensão com alto nível de heterogeneidade de formas geométricas no conjunto de itens, os tempos de execução observados não foram satisfatórios, devido ao alto nível de complexidade do algoritmo. Tornou-se claro que seria vantajoso, tendo em vista uma redução do tempo necessário para alcançar uma solução aceitável, explorar uma estratégia de alocação com base em algoritmos heurísticos em detrimento dos algoritmos exactos.

O conjunto de heurísticas estudadas e apresentadas na revisão bibliográfica, foi um factor determinante na escolha de uma heurística para a construção de uma solução para este problema. A alternativa escolhida foi a heurística de melhor encaixe, com corte livre e itens não orientados, que será apresentada no Capítulo 4.
Verificou-se a necessidade de adaptação da função objectivo do problema de empacotamento a duas dimensões típico (do modelo de Lodi et al. [7]), que consistia em minimizar o número de placas utilizadas. Esta abordagem permitia realizar uma análise comparativa satisfatória entre o modelo e os resultados iniciais, principalmente para instâncias grandes com um elevado nível de heterogeneidade de formas geométricas no conjunto de itens, comprovando-se a coerência dos resultados pela sua proximidade.

Todavia, foi observado que para um conjunto de instâncias iniciais com um número de itens reduzido, as soluções apuradas não apresentaram quaisquer diferenças entre si, resultando no mesmo número de placas final. Tornou-se portanto necessário aumentar o nível de especificação da solução, para despistar eventuais empates entre soluções equivalentes à partida, mas com diferenças qualitativas a um nível mais baixo.

A nova função objectivo de desempate, consiste em minimizar a altura da placa menos preenchida em altura. Deste modo, para duas soluções que apresentem um número de placas igual, é possível diferenciá-las qualitativamente através da comparação da altura da placa menos preenchida em altura de cada uma das soluções.

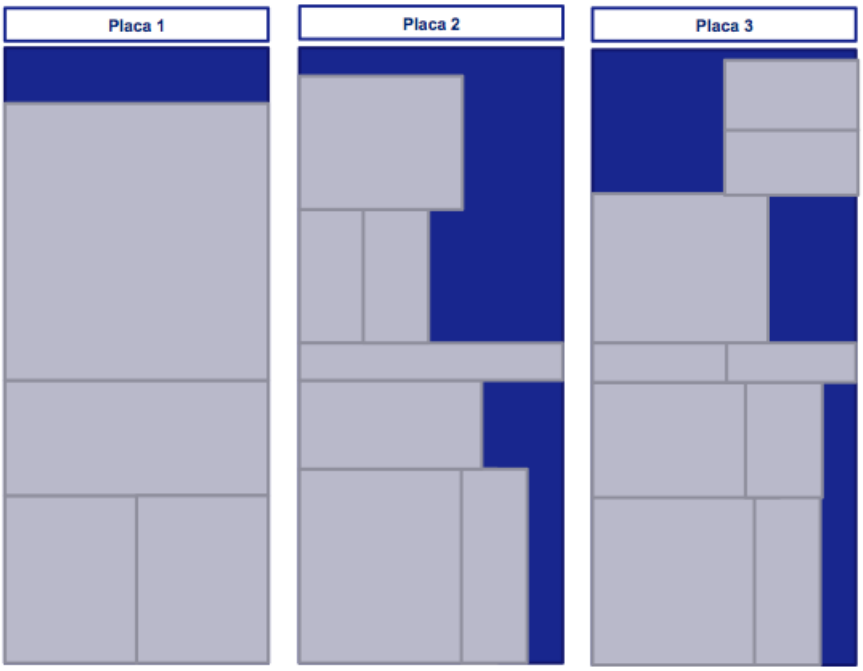

Figura 9: Função objectivo: Minimizar a altura da placa mais curta. Solução com pior qualidade.

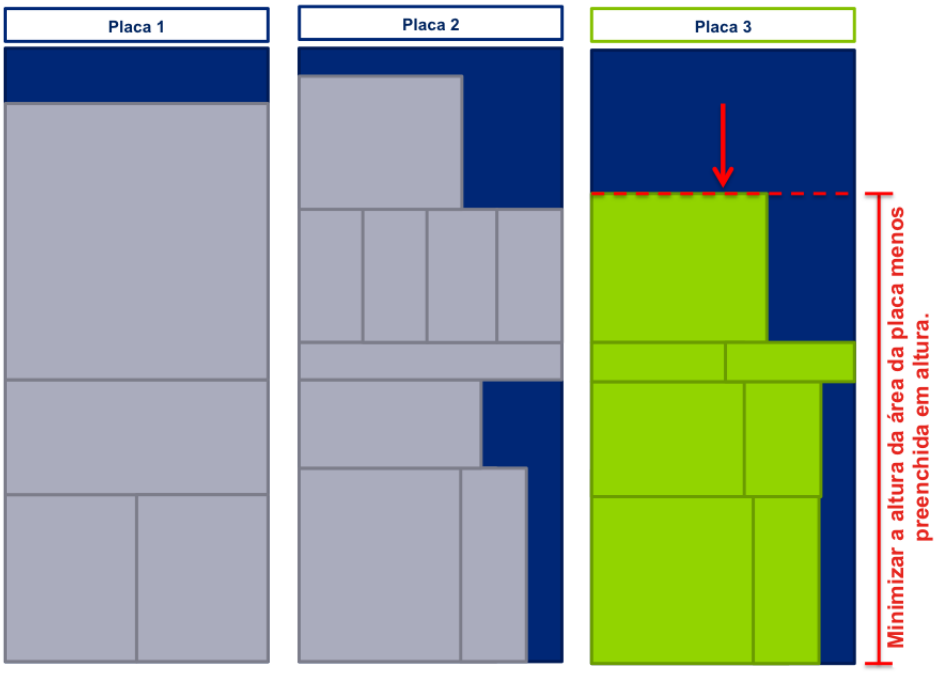

Figura 10: Função objectivo: Minimizar a altura da placa mais curta. Solução com maior qualidade.

A Figura 9 e a Figura 10 ilustram a alteração efectuada à função objectivo de desempate do problema de corte: enquanto que anteriormente, o resultado seria um total de 3 placas, na forma alterada o resultado final passa a ser a altura máxima da área preenchida da placa com menor altura, neste caso representado a verde. Através da diferença entre as figuras apresentadas, verificamos uma solução de maior qualidade no primeiro caso, por apresentar uma menor altura na placa com menor área de preenchimento. Minimizando a altura desta placa em específico, é automaticamente reduzida a área desperdiçada das restantes placas utilizadas.

4 Método de solução

4.1 Descrição

4.1.1 Heurística de melhor encaixe

O método apresentado por Burke et al. [2] é orientado a problemas de empacotamento em rolos, e realiza uma selecção dinâmica dos itens durante o estágio de empacotamento baseada na estratégia de melhor encaixe. Isso possibilita ao algoritmo realizar decisões informadas sobre qual o item a ser empacotado a seguir, e também sobre a sua localização. Imahori e Yagiura [4] realizaram uma análise ao trabalho de Burke et al. [2], realçando a simplicidade e a performance deste algoritmo, apresentando também uma implementação eficiente do mesmo.
O facto de o método de Burke et al. [2] seleccionar dinamicamente os itens a serem empacotados em cada fase de empacotamento através da pesquisa de melhor encaixe, ao contrário de alguns dos métodos analisados anteriormente que seleccionam os itens a empacotar sequencialmente após uma ordenação prévia à fase de empacotamento, foi um factor que contribuiu para a sua selecção. Outro factor determinante para a escolha deste método, foi o facto de possuir um conjunto de características positivas presentes em algumas das heurísticas analisadas, tais como a rotação de itens e o empacotamento livre. Foi também relevante a versatilidade do algoritmo, que ao conjungar diferentes políticas de empacotamento geradoras de diferentes soluções, selecciona sempre a melhor alternativa disponível para o problema solucionado.
Para determinar o espaço mínimo disponível (nicho), é necessário analisar o rolo e o padrão de itens empacotados. Assim que o nicho é definido, procede-se à pesquisa do item que j se encaixa nesse espaço.
No início do processo de empacotamento, não há qualquer item empacotado no objecto. Desta forma, o mínimo espaço disponível (nicho) para o empacotamento de um item, corresponde à largura total do objecto (W). À medida que os itens vão sendo alocados, o nicho vai sendo alterado consecutivamente, tanto em largura como em altura.
O item que melhor se adapta é sempre seleccionado, verificando-se três distintas possibilidades:

1. **Há um item não empacotado com uma das dimensões (largura ou altura) coincidente com a largura do nicho.** Neste caso, o empacotamento do item torna-se fácil havendo uma total compatibilidade do nicho. No caso em que vários itens são totalmente compatíveis com o espaço disponível, é escolhido o item com maior área.

2. **Há um item não empacotado com dimensões menores à largura do nicho**. É

seleccionado o item que ocupa a maior porção do nicho. Uma vez que o item não ocupa o nicho na totalidade, torna-se necessário definir a forma de colocação do item seguindo a política de posicionamento escolhida.

3. **Não há itens empacotados com dimensões compatíveis com a largura do nicho.** Neste caso o espaço disponível é considerado desperdício, já que não havendo no momento itens compatíveis, então nenhum item será compatível em iterações futuras.

Uma das vantagens da estratégia de melhor encaixe é o facto de apenas serem criados buracos de desperdício que não podem ser preenchidos por outros itens. Outra vantagem é o facto de ser assegurada a não sobreposição de itens aos padrões de empacotamento em cada fase de alocação

Políticas de preenchimento de nichos A heurística de melhor encaixe é uma combinação de três políticas distintas de preenchimento de nichos, sendo que cada política indica a forma de posicionamento dos itens quando estes não são totalmente compatíveis com a largura dos nichos.
Os nichos são criados apenas quando nenhum dos itens não empacotados é compatível com o nicho actual. Este problema considera apenas um objecto com largura finita e altura infinita, tratando-se portanto de um problema de empacotamento em rolos. As três políticas de preenchimento de nichos são:

1. **Posicionamento à esquerda -** A política de posicionamento à esquerda coloca itens parcialmente compatíveis nos nichos, alinhando-os à esquerda.

2. **Posicionamento junto ao vizinho mais alto -** Nesta política são examinados dois itens empacotados no objecto, que definem o nicho. É seleccionado o item a ser colocado no nicho através da pesquisa de melhor encaixe, junto ao maior dos dois itens que o definem. Se o nicho for definido por um item e por uma das extremidades laterais do objecto, o item é colocado junto à segunda. É possível visualizar um esquema desta política de empacotamento na Figura 11.

3. **Posicionamento junto ao vizinho mais baixo -** Esta política de posicionamento é oposta à política de posicionamento do vizinho mais alto descrita acima. Neste caso, o novo item é colocado a seguir ao item mais baixo.

Figura 11: Exemplo de posicionamento junto ao vizinho mais alto.

Uma desvantagem do método de Burke et al. [2] é a possibilidade da geração de soluções com má qualidade devido à ocorrência de **torres** de itens nos padrões de empacotamento. Este fenómeno sucede quando é realizado o empacotamento sucessivo de vários itens com altura elevado na parte final do processamento. Esta situação pode ser revertida, através da rotação de 90^{o} dos itens que geram as torres, possibilitando uma melhoria da solução final. Este processo de optimização da solução é realizado no final do processamento do algoritmo.

Representação da informação A informação do empacotamento é guardada num *array* linear (*array skyline*) com tamanho igual à largura do objecto, que consiste numa linha quebrada:

- Cada segmento da linha é horizontal.
- Duas linhas adjacentes têm coordenadas yy diferentes, e exactamente uma posição x comum (isto é, a coordenada x da extremidade direita de uma linha é igual à coordenada x da extremidade esquerda da outra linha).
- Vista de uma posição infinitamente alta, nenhum ponto das linhas é oculto por um item empacotado.
- Cada linha toca na aresta superior de um item alocado ou na base do rolo.
- De todas as linhas representadas no *array skyline*, a linha considerada como o espaço mínimo disponível (ou nicho) é a linha com coordenadas yy mais baixas.

Imahori e Yagiura [4] apresentaram uma proposta mais eficiente para a guardar a informação do *array skyline*, utilizando uma *heap* e uma lista ligada bi-direccional em vez do *array* linear. Também utilizaram uma árvore binária de procura para a pesquisa do item de melhor encaixe,

que guarda os itens não empacotados e os compara por valores de largura (w_i). A estrutura de dados utilizada no presente trabalho foi o *array* linear.
Cada elemento do *array* guarda a altura total do empacotamento de cada coordenada x do rolo. A coordenada do nicho pode ser encontrada localizando o menor valor guardado no *array*, e a sua largura é calculada pelo número de posições consecutivas de igual valor.

Pesquisa de melhor encaixe Os dados dos itens são definidos numa lista de pares (w_i, h_i) com os valores de largura e de altura. Para facilitar a pesquisa dos itens durante o processamento do algoritmo, os itens são orientados horizontalmente, e depois ordenados por valores não crescentes de largura (em caso de empate, são ordenam-se por valores não crescentes de altura). O resultado da pesquisa de melhor encaixe sobre um conjunto de itens com as propriedades mencionadas, devolve o item com maior aproximação às dimensões limite parametrizadas.

O algoritmo de melhor encaixe pode ser definido em pseudo-código da seguinte forma:

Algoritmo de melhor encaixe

1. Obter largura do rolo;
2. Obter lista de n itens;
3. Orientar itens horizontalmente ($w_j \geq h_j, j \in J$), ou seja, atribuindo à dimensão de largura a aresta com maiores dimensões, e à dimensão de altura a aresta com menores dimensões;
4. Ordenar lista de itens por ordem decrescente de largura (desempatando por valores decrescentes de altura);
5. Inicializar *array skyline* de n elementos.
6. **Para cada** política de posicionamento (à esquerda, vizinho mais alto, vizinho mais baixo) **executar**
   (a) **enquanto** (houver itens por empacotar)
      i. Calcular dimensões do nicho.
      ii. **Se** encontrado item compatível
      então empacotar item seguindo política de posicionamento; Actualizar array;
      caso contrário Gerar área desperdiçada junto ao item vizinho mais baixo.
   (b) Retornar melhor solução.

4.1.2 Uma modificação da heurística de melhor encaixe

A escolha da heurística a ser implementada na ferramenta de planeamento de corte recaiu sobre o algoritmo de melhor encaixe. A heurística proposta por Burke et al. [2] orientada para problemas de empacotamento em rolos, tal como foi descrito anteriormente, realiza a selecção dinâmica dos itens durante a fase de alocação, seguindo a estratégia de alocação de melhor encaixe em rolos. Para tornar possível a aplicação desta heurística ao caso *MPinto*, impôs-se a necessidade de adaptar o algoritmo de uma lógica de empacotamento em rolos a uma lógica de problemas de empacotamento em placas. Se na versão para empacotamento em rolos tínhamos um problema de empacotamento a duas dimensões com apenas um objecto, o

novo algoritmo irá alocar os itens sucessivamente num conjunto finito de placas conservando a mesma estratégia de preenchimento. O resultado gerado pela heurística foi também adaptado à nova função objectivo adoptada, ou seja, a minimização da altura da área da placa menos preenchida em altura.

4.1.3 *Array Skyline*

O processo de empacotamento de itens em placas é controlado através de um *array* chamado *skyline*, cuja dimensão é equivalente à largura do objecto em centímetros. Sempre que é inicializada uma nova placa para ser preenchida, o respectivo *array skyline* é criado, com todos os elementos a zero, indicando que o objecto se encontra vazio. É um elemento essencial no processo de cálculo do nicho de alocação, e na procura do item de melhor encaixe, uma vez que representa as coordenadas da superfície de alocação da placa.

Sempre que é efectivada a alocação de um item na placa, o *array skyline* é actualizado, reflectindo a alteração na altura da área preenchida no objecto. O exemplo da Figura 12 ilustra o processo de actualização do *array skyline* associado à alocação de itens.

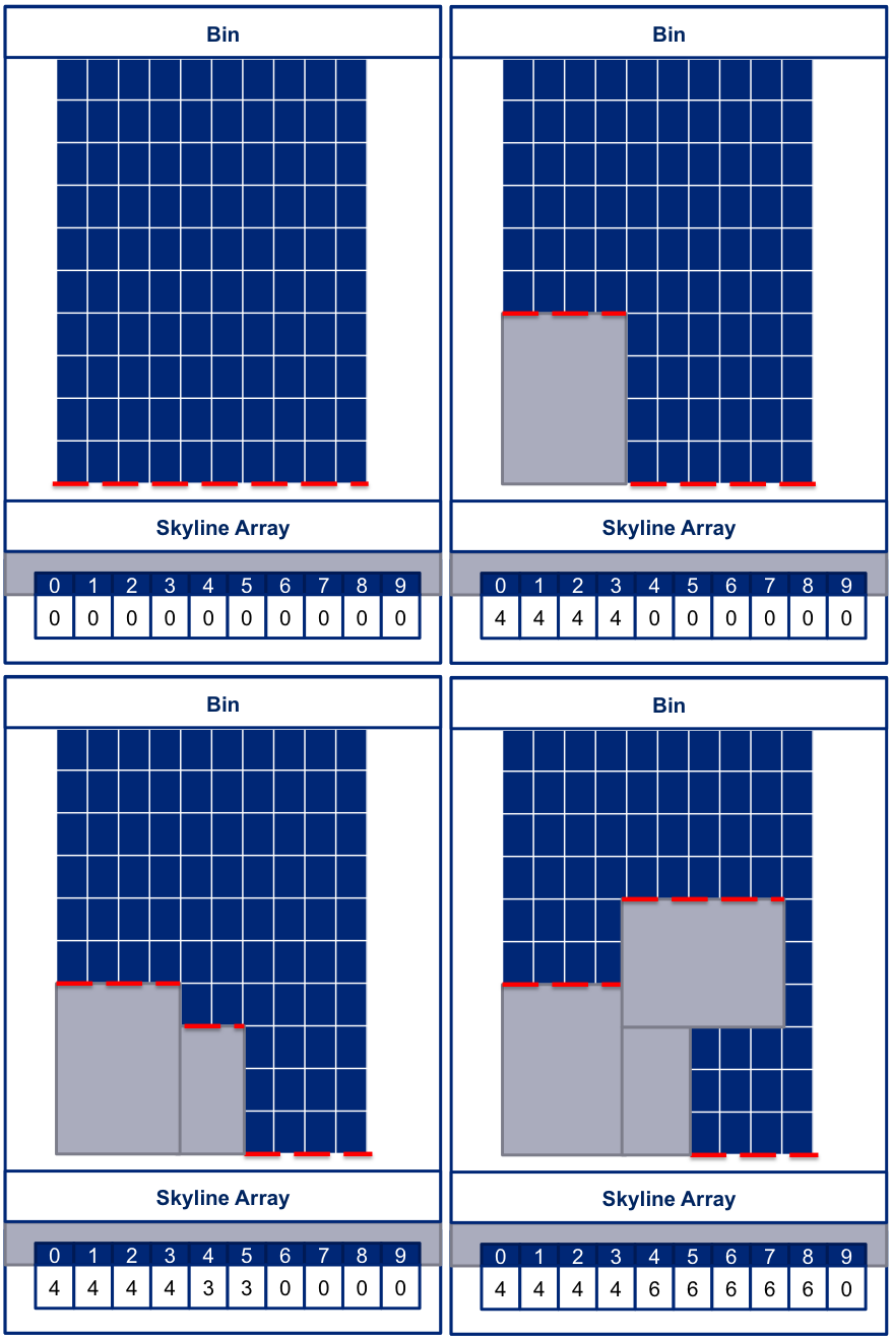

Figura 12: Actualização do *array skyline*.

4.2 Algoritmo

A heurística começa por importar os parâmetros de entrada. Sequencialmente, é executado o algoritmo de melhor encaixe para as três políticas de posicionamento existentes, que estão representadas na Figura 13:

- Posicionamento à esquerda
- Posicionamento ao lado do vizinho mais alto
- Posicionamento ao lado do vizinho mais baixo

Figura 13: Políticas de posicionamento de itens.

Para um conjunto não nulo finito de itens, são preenchidas n placas. No início da alocação ($i = 0$), é inicializada a $placa_i$ e o respectivo $skyline_i$. Até que todas as posições do $skyline_i$ atinjam a altura máxima da $placa_i$, isto é, até que não haja nenhum item compatível com o espaço disponível da $placa_i$ ao ponto de a área livre ser transformada em área desperdiçada, são alocados sucessivamente os itens de melhor encaixe correspondentes ao nichos calculados em cada iteração. Os nichos, também designados por espaços mínimos disponíveis, correspondem aos segmentos do *array skyline* com coordenada yy menor disponível em cada iteração.

Depois de um determinado nicho ser calculado, é realizada a pesquisa de melhor encaixe ao conjunto de itens não empacotados. Se for encontrado o melhor item compatível é empacotado directamente se este tiver uma das dimensões coincidente com as dimensões do nicho, ou seguindo a política de empacotamento em vigor caso as dimensões sejam menores que as dimensões do nicho. Caso não seja encontrado nenhum item compatível, o nicho é transformado em área não preenchida desperdiçada. No fim do empacotamento é actualizado o *array skyline*. Os resultados possíveis da pesquisa *best-fit* estão representados na Figura 14:

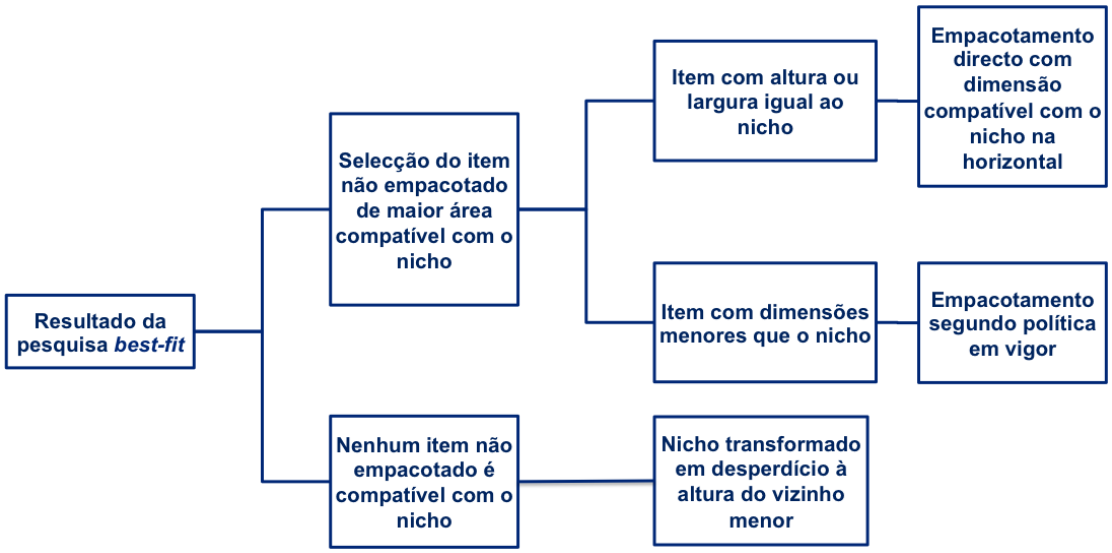

Figura 14: Resultados possíveis da pesquisa de melhor encaixe.

O algoritmo implementado é ilustrado na Figura 15.

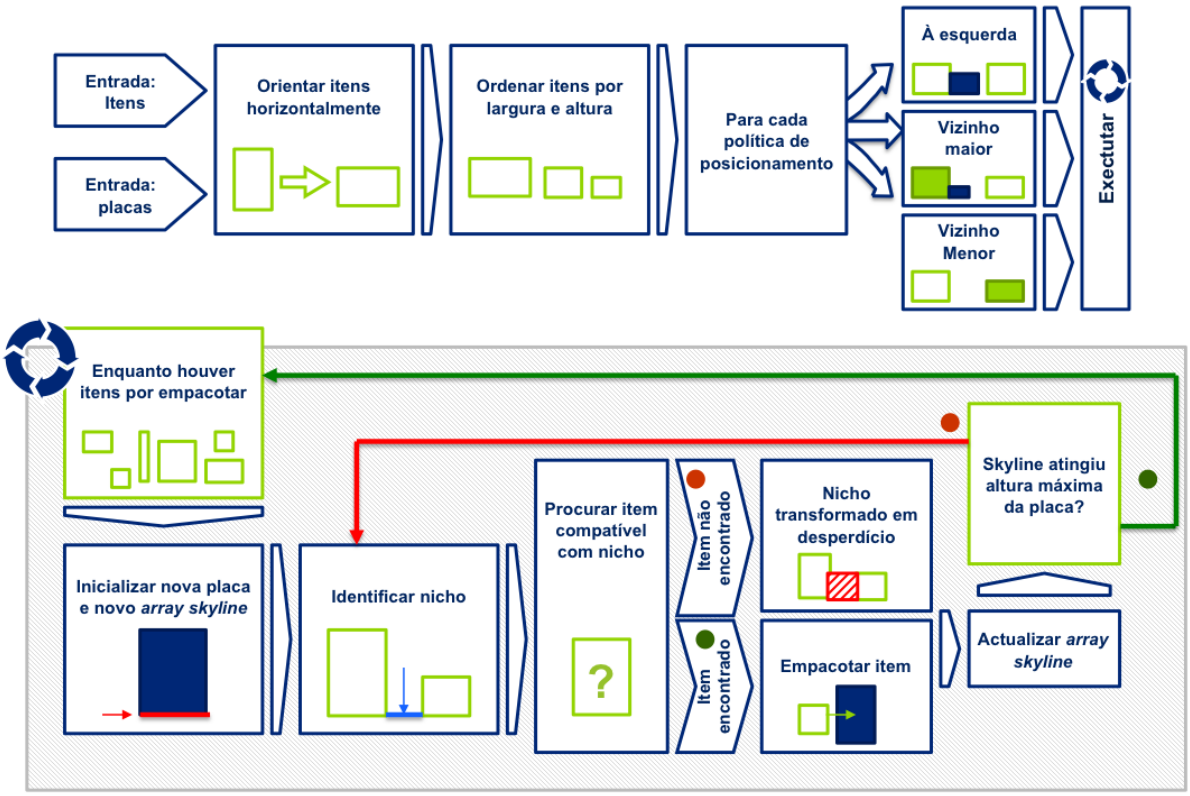

Figura 15: Diagrama da heurística de melhor encaixe adaptada a problemas de empacotamento em placas.

Quando a execução da heurística atinge um ponto em que todas as posições do *array skyline* atingem o valor da altura da placa, é realizada uma nova iteração do ciclo, sendo aberta uma nova placa com o respectivo *array*. Se o conjunto de itens não empacotados estiver vazio, é terminada a fase de empacotamento.

O resultado devolvido pelo algoritmo traduz o número de placas utilizadas na solução do problema, especificando para cada uma delas a altura máxima da área preenchida. Com base nesta informação, é possível averiguar a solução final da função objectivo, escolhendo o valor mínimo da altura máxima da área preenchida do conjunto de placas da solução.

4.3 Exemplo

Para uma melhor percepção do funcionamento do algoritmo implementado, é apresentado o resultado de execução de uma instância de teste.

Consideremos a instância de corte a duas dimensões representada na Tabela 7.

Tabela 7: Instância para a extensão à heurística de melhor empacotamento

Indíce i	Código c_i	Altura h_i	Largura w_i
1	A	3,6	1,4
2	B	1,4	1,4
3	C	2,0	0,2
4	D	1,3	1,8
5	E	2,0	2,0
6	F	1,0	3,0
7	G	1,2	2,6
8	H	0,1	1,4

Sejam as dimensões dos objectos da instância:

- $W = 4$
- $H = 8$

Após a orientação horizontal do conjunto de itens e da sua ordenação por valores não crescentes de largura e altura, é obtida a disposição de itens na Tabela 8.

Tabela 8: Orientação horizontal e ordenação de itens.

Indíce i	Código c_i	Altura h_i	Largura w_i
1	A	1,4	3,6
6	F	1,0	3,0
7	G	1,2	2,6
5	E	2,0	2,0
3	C	0,2	2,0
4	D	1,3	1,8
2	B	1,4	1,4
8	H	0,1	1,4

A heurística inicia a sua execução adoptando a política de posicionamento à esquerda. Após a execução completa, realizam-se duas novas execuções adoptando respectivamente a política de posicionamento junto ao vizinho mais baixo, e junto ao vizinho mais alto.

No início do processo de empacotamento, inicializa-se o *array skyline* com $W * 100$ posições, sendo que cada unidade corresponde a cada centímetro da largura total da placa. Neste caso o tamanho do *array* é 400, e no início da execução da heurística todas as posições estão a zero. É identificado o nicho inicial para se realizar o empacotamento. Uma vez que a placa está vazia, o nicho corresponde à base da placa.

A pesquisa de melhor encaixe procura o subconjunto de itens por empacotar com dimensões compatíveis com o nicho definido na presente iteração. Na iteração inicial todos os itens são compatíveis com o nicho, sendo por isso seleccionado e empacotado o item com maior aproveitamento do espaço, que neste caso é o item A.

Após o empacotamento do item A na posição (0,0) da placa, o *array skyline* é actualizado com os valores da altura do item A nas posições $[0; w_A]$. Uma vez que a altura do *array skyline* não atingiu a altura máxima da placa, é definido um novo nicho para realizar um novo

empacotamento na mesma placa.

O novo nicho definido corresponde ao espaço à direita do item A, gerado pelo segmento do *array skyline* com menor cota alinhado à esquerda. Uma nova pesquisa de melhor encaixe identifica os itens F e H com orientação vertical como sendo compatíveis com o nicho. Sendo a política de empacotamento em vigor à esquerda, é empacotado o item F à direita do item A, por apresentar um maior aproveitamento do espaço disponível, e subsequentemente é empacotado o item H.

Após o empacotamento do item H, não havendo itens compatíveis com o nicho definido, a área não empacotada à direita do item H é desperdiçada. A altura do *array skyline* nas posições correspondentes ao nicho desperdiçado, é preenchido com o valor da altura do item mais próximo, neste caso o item H com orientação vertical.

É processado o empacotamento dos restantes itens até que a altura do *array skyline* atinja a altura máxima da placa. Quando a altura máxima da placa é atingida, é inicializada uma nova placa para se empacotarem os restantes itens. Assim que todos os itens são empacotados, é terminada a execução da heurística.

A sequência de empacotamento para a política de posicionamento à esquerda é:

$$A, C, H, F, G, B, E, D$$

Após a execução da heurística com a instância apresentada, o resultado gerado pela ferramenta é apresentado no gráfico da Figura 16.

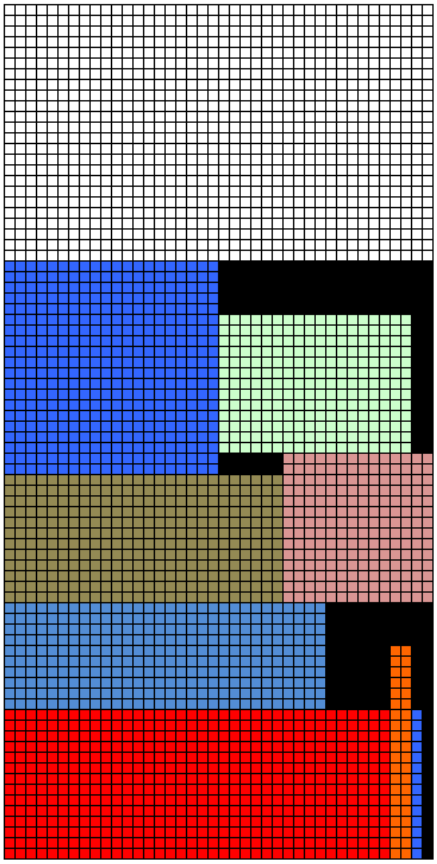

Figura 16: Resultado da execução da instância para a extensão da heurística de melhor empacotamento com posicionamento à esquerda.

Para esta instância o resultado das três políticas de empacotamento é equivalente, uma vez que a altura da área empacotada da placa é igual em todos os casos. Na Figura 17 é apresentado um esquema do resultado da heurística para as duas políticas de empacotamento restantes: posicionamento ao lado do vizinho mais baixo, e posicionamento ao lado do vizinho mais alto.

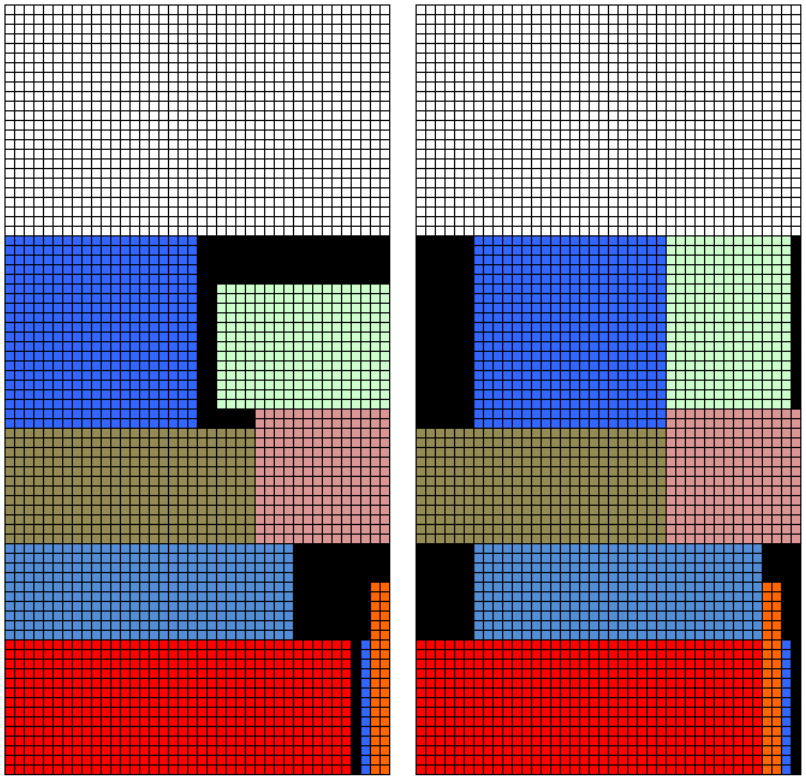

Figura 17: Resultado da execução da instância para a extensão da heurística de melhor encaixe com posicionamento ao lado do vizinho mais baixo, e posicionamento ao lado do vizinho mais alto.

5 Implementação e resultados experimentais

5.1 Implementação de modelos 2LBP e 2LBPr

Numa primeira fase do desenvolvimento do presente projecto, foi construída uma ferramenta de planeamento de corte a partir do modelo 2LBP proposto por Lodi et al. [7], com o auxílio da tecnologia *COIN-OR CBC* [5]. O *CBC (COIN-OR Branch-and-Cut)* é uma ferramenta *solver* de *software* livre, de programação inteira construída em C++, que potencializou a execução de modelos de programação linear com as instâncias da *MPinto*, nomeadamente os modelos 2LBP e 2LBPr.

As soluções geradas a partir dos modelos 2LBP e 2LBPr pertencem à classe de problemas 2BP guilhotinável a duas fases (empacotamento por níveis), com e sem rotação de itens, respectivamente. Uma vez que as soluções geradas pela *MPinto* foram na sua maioria classificadas como soluções para problemas de empacotamento de níveis e pilhas, a qualidade das soluções obtidas pelos modelos 2LBP e 2LBPr revelou-se consequentemente pior do que a das primeiras.

O ponto forte dos dois modelos implementados, principalmente do 2LBPr que adiciona a capacidade de rodar itens ao modelo 2LBP, é possibilitar a convergência da solução para o valor óptimo do modelo de programação inteira. Todavia, se por um lado a qualidade das soluções fica à partida comprometida pelo facto de o modelo só gerar soluções de corte guilhotinável a duas fases, por outro lado o tempo de execução do modelo cresce exponencialmente para instâncias com um número elevado de itens com um grande nível de heterogeneidade de dimensões.

É apresentado em anexo um esquema de uma solução obtida com o modelo 2LBP, comparada com a solução homóloga da *MPinto* para a mesma instância, nas Figuras 18 e 19.

5.2 Extensão de melhor encaixe

O algoritmo *best-fit* de Burke et al. [2] foi implementado com o objectivo de ultrapassar, tanto em tempo de processamento com em quantidades de matéria-prima utilizada, os resultados alcançados com o modelo de Lodi et al. [7]. A heurística descrita anteriormente na secção 'Heurística de melhor encaixe' (Secção 4.1.1), sofreu as alterações necessárias para

ser adaptada a problemas de empacotamento em placas, e o mecanismo de controlo de posicionamento, o *array skyline*, que passou a ser replicado por cada placa aberta (Secção 4.1.2).

A implementação foi efectuada em C++, tendo sido utilizada a biblioteca de domínio público (*open source*) da autoria de Jukka Jylänky [6], que veio capacitar a ferramenta construída com algumas estruturas e funções que estão na base do processo de empacotamento implementado.

A exportação do resultado do algoritmo é realizado para um ficheiro de texto, onde é especificado o número de placas resultantes, bem como altura máxima da área preenchida de cada placa, que constitui a informação relevante para apurar a solução da função objectivo. Também é exportada a informação da disposição dos itens nas placas onde foram alocados, em formato LaTeX. Os ficheiros LaTeX ao serem compilados fornecem uma visualização gráfica do plano de corte de matéria-prima, que poderá fornecer um auxílio de grande utilidade na fase de aplicação do material no local de obra. É possível visualizar um exemplo ilustrativo em anexo neste documento, na Figura 20.

5.3 Resultados de testes a modelos 2LBP e 2LBPr

Foram realizados testes aos modelos 2LBP e 2LBPr implementados no $COIN - OR\ CBC$ com as instâncias que foram fornecidas inicialmente pela *MPinto*.

O sistema utilizado para os testes computacionais contém as seguintes características:

- Processador: 2.5 GHz Intel Core 2 Duo
- Memória: 2GB 667 MHz DDR2 SDRAM

Os testes foram realizados em duas fases distintas, com tempos de execução máximo diferentes:

- **Fase 1 -** Tempo máximo de execução: 30 min.
- **Fase 2 -** Tempo máximo de execução: 2 horas.

A necessidade da realização de testes com tempos máximos de execução mais extensos impôs-se ao ser verificado que, para algumas instâncias, o valor óptimo não era alcançado com testes de 30 minutos de execução. Nos casos em que o valor óptimo foi alcançado na primeira fase

de testes, não foram repetidos os testes na segunda fase. Foi também comprovado que para algumas instâncias, apesar de o processo de execução não chegar ao fim durante o período máximo de processamento, é possível atingir uma solução óptima, como é o caso da instância *MEDINTONE_885-310* e *MEDINTONE_885-302*.

As instâncias *Quantidades (Small)* e *Quantidades (Big)* são sub-instâncias da instância *Quantidades (Full)*, criadas especificamente para os testes dos modelos, e que não foram submetidas a testes por parte da *MPinto*. Por este motivo, não têm resultados aplicáveis (-) na coluna $placas_{MPinto}$.

Nos casos em que nas colunas para execução com tempo máximo de 2 horas não temos valores preenchidos (-), estamos perante instâncias que obtiveram valores óptimos nas execuções com tempo máximo de 30 minutos. Quando o tempo de execução do modelo chega ao limite máximo sem que tenha sido encontrada a solução óptima, é devolvida a solução de maior qualidade encontrada até esse momento.

Podemos observar os resultados obtidos para o modelo 2LBP e 2LBPr nas Tabelas 9 e 10, respectivamente. Em cada uma das tabelas mencionadas, temos as seguintes colunas:

- Instância - Nome da instância submetida à execução do modelo em teste.
- n_{itens} - Número de itens da instância a serem empacotados.
- P_{MPinto} - Número de placas da solução obtida pela *MPinto* para a instância analisada.
- $t_{max:30m}$ - Tempo de execução (em segundos) do modelo para um tempo máximo de execução de 30 minutos.
- $P_{max:30m}$ - Número de placas devolvido pela execução do modelo para um tempo máximo de execução de 30 minutos.
- $t_{max:2h}$ - Tempo de execução (em segundos) do modelo para um tempo máximo de execução de 2 horas.
- $P_{max:2h}$ - Número de placas devolvido pela execução do modelo para um tempo máximo de execução de 2 horas.

Tabela 9: Resultados de execução do modelo 2LBP

Instância	n_{itens}	P_{MPinto}	$t_{max:30m}$	$P_{max:30m}$	$t_{max:2h}$	$P_{max:2h}$
Quantidades (Full)	31	3	22	3	-	-
Quantidades (Small)	12	-	1	1	-	-
Quantidades (Big)	19	-	3	3	-	-
MEDINTONE_885-310	135	54	1828	59	7264	59
MEDINTONE_885-303	9	8	4	8	-	-
MEDINTONE_885-302	102	26	1834	29	7402	28

Tabela 10: Resultados de execução do modelo 2LBPr

Instância	n_{itens}	P_{MPinto}	$t_{max:30m}$	$P_{max:30m}$	$t_{max:2h}$	$P_{max:2h}$
Quantidades (Full)	31	3	25	3	-	-
Quantidades (Small)	12	-	47	1	-	-
Quantidades (Big)	19	-	10	3	-	-
MEDINTONE_885-310	135	54	1900	58	7304	58
MEDINTONE_885-303	9	8	14	8	-	-
MEDINTONE_885-302	102	26	*	*	7288	30

(*) Nenhuma solução obtida no tempo limite.

(-) Não aplicável.

5.4 Heurística de melhor encaixe para um problema de empacotamento a duas dimensões

Para a fase de testes da construção da ferramenta de planeamento de corte deste projecto, foi reunido um conjunto de instâncias reais pela empresa *MPinto*, consideradas como casos modelo de aplicação prática, capazes de fornecer informação relevante sobre a performance da ferramenta.

Foram realizados testes à heurística de melhor encaixe modificada (secção 4.1.2) para um problema de empacotamento a duas dimensões com um total de 31 instâncias, com conjuntos de itens de cardinalidade variável entre 9 e 136 elementos.

Os resultados dos mesmos testes são apresentados na Tabela 11.

Tabela 11: Resultados da heurística de melhor encaixe para um problema de empacotamento a duas dimensões

Instância	n_{itens}	W	H	P_{MPinto}	P_{me}	$minH_{MPinto}$	$minH_{me}$	t_{me}
MEDINTONE_885-303	9	183	250	8	8	10.73	10.73	0.138
ALQ_200_100	9	200	205	4	5	20.50	13.15	0.086
TORRES_VEDRAS	9	200	230	2	2	14.39	17.76	0.056
ENFERMAGEM_33445	14	200	250	6	8	12.99	12.99	0.129
PR_8202	15	200	250	2	2	14.77	15.30	0.051
URGESES	18	200	250	5	5	7.13	21.30	0.117
PR_33637-AZUL	21	200	250	3	4	24.79	9.94	0.105
PR_17102-SMOKE	23	200	250	3	3	2.20	3.95	0.106
COR_JESUS	24	200	250	5	5	5.10	17.00	0.127
JF_BARCELOS_VFSP	24	200	250	3	3	16.82	20.64	0.103
PR_33602-CINZA+	24	200	250	3	3	14.93	20.84	0.108
TEBOSA	26	200	200	6	7	13.45	11.59	0.173
AMP_LCS_SOBREPOSTA	28	200	250	4	5	24.84	15.40	0.136
PR_33635-AMARELO	30	200	250	4	4	2.45	5.09	0.144
ENFERMAGEM_33441	31	200	230	12	13	10.70	19.09	0.362
QUANTIDADES (FULL)	31	200	230	3	3	19.24	18.09	0.126
AMP_JF_SOBREPOSTA	33	200	250	5	5	10.92	18.75	0.148
PR_33656-ROSA	42	200	250	3	3	19.50	21.50	0.186
FACHA	51	183	250	8	9	6.39	8.02	0.568
PR_33638-VERDE	51	200	250	4	4	19.85	18.90	0.243
MURTOSA_WALL	54	200	200	2	2	4.00	19.14	1.093
PR_33605-BEJE	54	200	250	3	4	22.73	6.45	0.222
ENFERMAGEM_33440	57	200	250	9	10	6.94	10.82	0.288
PR_33603-CREME	68	200	250	5	5	16.80	22.40	0.348
ES_BARCELOS_ALCAT	76	200	300	12	15	20.28	17.85	0.484
PR_17100-FOG	81	200	250	5	5	23.17	20.75	0.428
ATL_ARRIFES	93	200	230	15	18	21.45	15.50	0.462
MURTOSA_FLOOR	100	200	200	29	35	8.75	13.60	0.18
MEDINTONE_885-302	102	183	250	26	31	15.07	17.14	0.73
PR_33630-CINZA	105	200	250	8	8	7.91	12.65	0.665
MEDINTONE_885-310	136	183	250	54	68	23.50	12.70	1.325

Legenda da Tabela 11

Instância - Nome da instância submetida à execução da heurística de melhor encaixe.

n_{itens} - Número de itens.

W - Largura das placas (em centímetros).

H - Altura das placas (em centímetros).

P_{MPinto} - Número de placas usado na solução da *MPinto*.

P_{me}- Número de placas usado na solução obtida com a heurística de melhor encaixe.

$minH_{MPinto}$ - Altura da placa com menor preenchimento em altura na solução da MPinto (em centímetros).

$minH_{me}$ - Altura da placa com menor preenchimento em altura na solução da heurística de melhor encaixe (em centímetros).

t_{me} - Tempo de execução da heurística de melhor encaixe (em segundos).

Estes resultados mostram que as soluções obtidas pela Mpinto são, em geral, de melhor qualidade que as obtidas pela heurística de melhor encaixe proposta. A análise dos dados dos resultados permitiu concluir que em média, as soluções originais gastam menos 10% de matéria-prima do que as soluções geradas pela heurística de melhor encaixe.

Por outro lado, os tempos de execução da heurística para o conjunto de instâncias de teste é aproximadamente 1 segundo, o que comparativamente ao tempo de preparação das soluções iniciais, são significativamente inferiores.

A excepção são as instâncias $QUANTIDADES(FULL)$, $PR\_33638-VERDE$ e $PR\_17100-FOG$ para as quais, embora o número de placas seja igual, a solução dada pela heurística de melhor encaixe modificada tem uma menor altura na placa com menor altura. É de salientar a rapidez da heurística de melhor encaixe modificada, apenas em duas instâncias demorou mais de um segundo a obter a solução.

6 Conclusões e trabalho futuro

Nesta dissertação apresentou-se uma heurística de melhor encaixe para empacotamento a duas dimensões, que está na base do desenho da ferramenta de optimização industrial construída. Através dos resultados obtidos na fase de testes da heurística de melhor encaixe para um problema de empacotamento a duas dimensões, é possível verificar que o tempo de execução do processo de planeamento de corte é instantâneo (com tempos de execução de aproximadamente 1 segundo) quando comparado com o caso *MPinto*, em que o processo de estereotomia efectuado manualmente envolve três colaboradores a tempo inteiro, durante vários dias de trabalho (com variações de 5 a 15 dias úteis, segundo fontes da empresa).

Os resultados revelaram níveis de qualidade assimétricos comparativamente às soluções originais, com baixa dependência do número de itens associado à instância, uma vez que as diferenças de qualidade das soluções não acompanham as diferenças de números de itens das instâncias. O grau de heterogeneidade de dimensões dos itens é um factor determinante nos resultados.
Apesar de a quantidade de matéria-prima consumida em cada instância ser excedida residualmente na maioria dos casos quando o planeamento é realizado com o auxílio da ferramenta, os custos associados à produção são reduzidos proporcionalmente a uma escala muito superior no que diz respeito ao número de horas-homem gasto em cada projecto.

A coerência e a constância no nível de qualidade dos resultados entregues aos clientes são assegurados pela objectividade e uniformização das soluções que a heurística produz, evitando a subjectividade e a exposição ao erro humano que são inerentes ao processo de planeamento manual. Com o uso deste algoritmo, a equipa de colaboradores tem a possibilidade de aumentar a sua produtividade, ganhando disponibilidade para se dedicarem a mais projectos em simultâneo.

Existem várias vertentes que podem ser exploradas no futuro para um desenvolvimento mais extensivo da ferramenta de planeamento de corte a duas dimensões:

- **Interface gráfico** - No início do projecto foi desde logo delineado que o âmbito da actividade de desenvolvimento não envolveria a construção de um interface gráfico, estando essa actividade prevista para uma fase posterior. É notória a necessidade de capacitar a

ferramenta construída de um ambiente amigável para o utilizador (neste caso, o técnico de estereotomia), capaz de fornecer auxílio na importação de dados de *input*, visualização de resultados, manipulação de parâmetros, ajustes manuais, edição de *output*, entre outros.

- **Integração com ferramentas de planeamento** - A exploração de um interface capaz de integrar vários *softwares* de corte e empacotamento, poderá possibilitar a utilização integrada desta ferramenta com outros sistemas já existentes baseados em algoritmos de naturezas distintas.

- **Integração com ferramentas CAD** - A utilização do *software* CAD é realizada na maioria dos projectos de estereotomia existentes, representando um papel fundamental na fase de identificação do conjunto de itens. A tarefa de determinação das áreas de corte, é efectuado com o auxílio das funcionalidades de cálculo de áreas do *software*. A capacitação de uma ferramenta de corte com funcionalidades integradas de cálculo de áreas automático a partir de ficheiros CAD, seria uma característica que traria valor acrescentado ao projecto, garantindo indubitavelmente uma optimização da produtividade da empresa.

- **Reaproveitamento de sobras de matéria-prima** - O desenvolvimento de um modelo de corte pertencente à classe de problemas de empacotamento em objectos com dimensões heterogéneas, poderá possibilitar o reaproveitamento automático de sobras existentes em *stock*, provenientes de projectos anteriores.

Referências

[1] M. A. Boschetti and A. Mingozzi. The two-dimensional finite bin packing problem. part ii: New lower and upper bounds. *4OR: A Quarterly Journal of Operations Research*, 1(2):135–147, 2003.

[2] E. K. Burke, G. Kendall, and G. Whitwell. A new placement heuristic for the orthogonal stock-cutting problem. *Operations Research*, 52(4):655–671, 2004.

[3] H. Dyckhoff. A typology of cutting and packing problems. *European Journal of Operational Research*, 44:145–159, 1991.

[4] S. Imahori and M. Yagiura. The best-fit heuristic for the rectangular strip packing problem: An efficient implementation and the worst-case approximation ratio. *Computers & Operations Research*, 37(2):325–333, 2010.

[5] R. L.-H. John Forrest. Cbc user guide. *http://www.coin-or.org/Cbc/cbcuserguide.html*, Web.

[6] J. Jylänki. Rectangle bin packing (open source library). *http://clb.demon.fi/rectangle-bin-packing*, Web.

[7] A. Lodi, S. Martello, and M. Monaci. Two-dimensional packing problems: A survey. *European Journal of Operational Research*, 141:241–252, 2002.

[8] A. Lodi, S. Martello, and D. Vigo. Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. *INFORMS Journal on Computing*, Fall vol. 11(4):345–357, 1999.

[9] A. Lodi, S. Martello, and D. Vigo. Models and bounds for two-dimensional level packing problems. *Journal of Combinatorial Optimization*, 8(3):363–379, 2004.

[10] J. Puchinger and G. Raidl. Models and algorithms for three-stage two-dimensional bin packing. *European Journal of Operational Research*, 183:1304–1327, 2007.

[11] G. Wäscher, H. Haußner, and H. Schumann. An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183:1109–1130, 2007.

7 Anexos

Instância: "Quantidades" (*MPinto*)

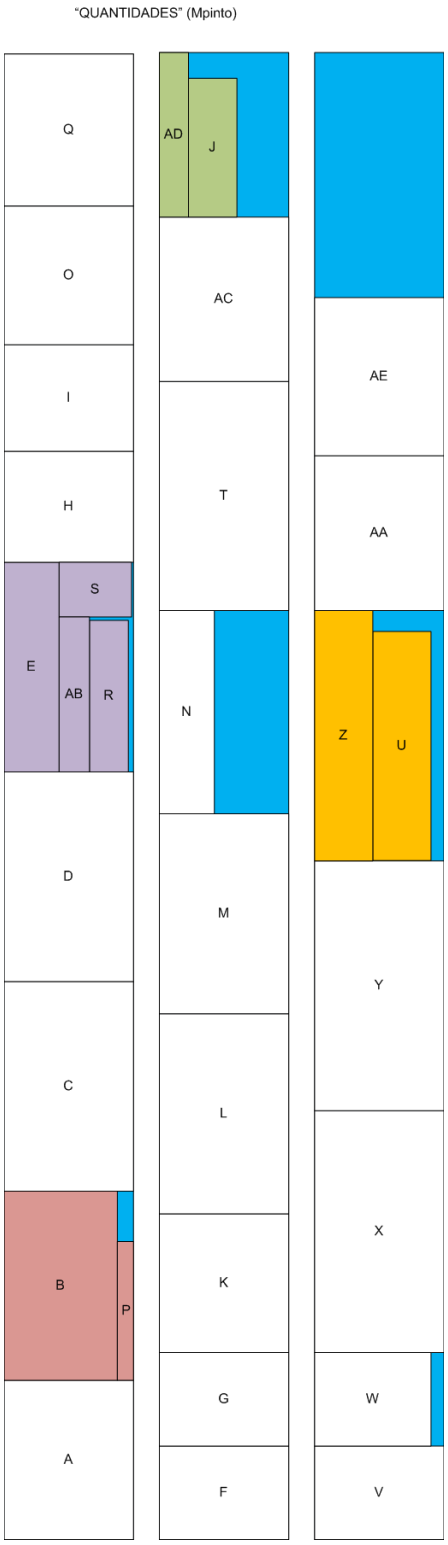

Figura 18: Instância: "Quantidades" - *MPinto*.

Instância: "Quantidades" (2LBP)

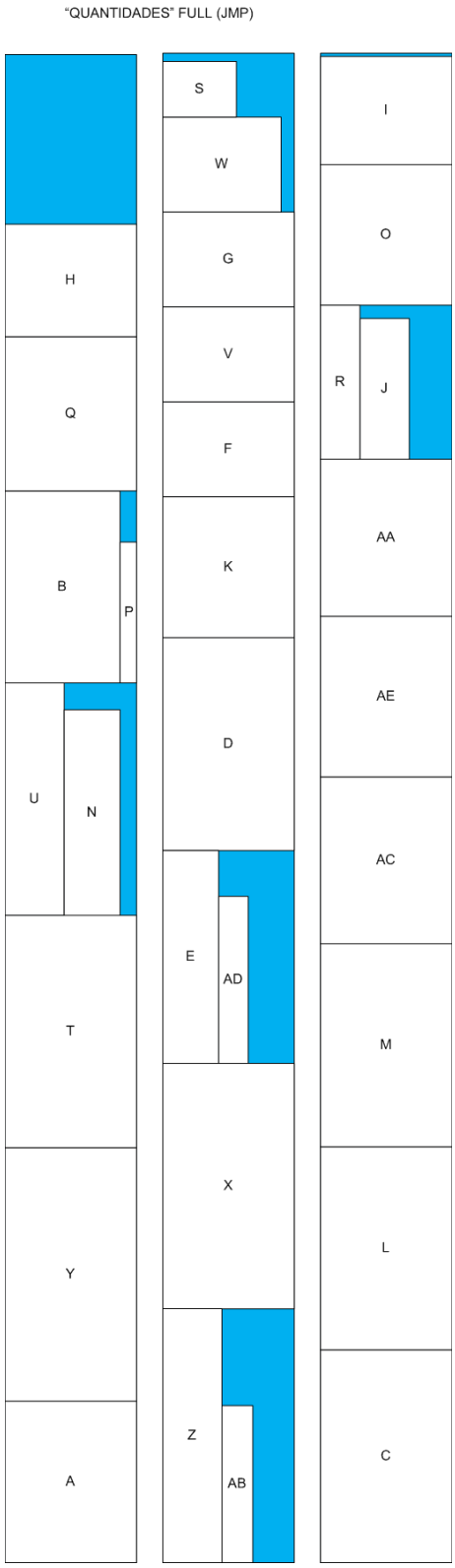

Figura 19: Instância: "Quantidades" - *2LBP*.

Instância: "Quantidades" (best-fit 2BP)

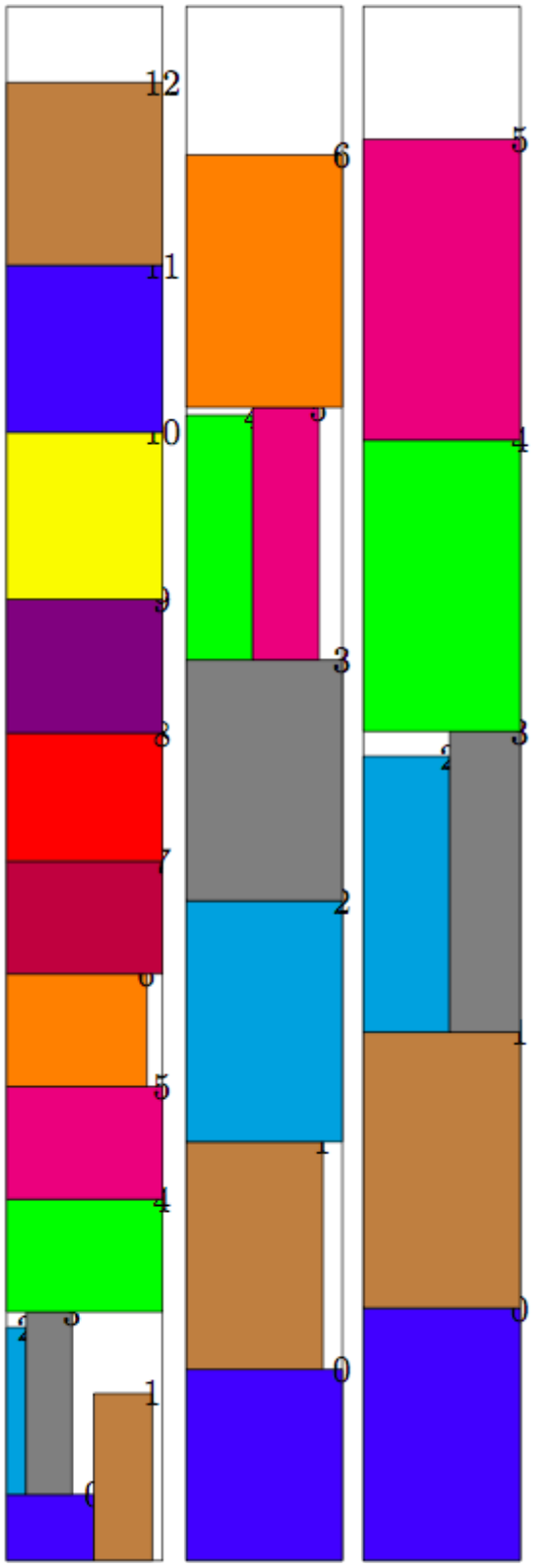

Figura 20: Instância: "Quantidades" - *best-fit 2BP*.